AN INTERACTIVE SYSTEM OF COMPUTER GENERATED GRAPHIC DISPLAYS FOR MOTIVATING MEANINGFUL LEARNING OF MATRIX OPERATIONS AND CONCEPTS OF MATRIX ALGEBRA

THESIS

Stone W. Hansard, Captain, USAF

AFIT/GSM/ENC/90S-12

DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

2

AFIT/GSM/ENC/90S-12

AN INTERACTIVE SYSTEM OF COMPUTER GENERATED GRAPHIC DISPLAYS FOR MOTIVATING MEANINGFUL LEARNING OF MATRIX OPERATIONS AND CONCEPTS OF MATRIX ALGEBRA

THESIS

Stone W. Hansard, Captain, USAF

AFIT/GSM/ENC/90S-12

The opinions and conclusions in this paper are those of the author and are not intended to represent the official position of the DOD, USAF, or any other government agency.

AFIT/GSM/ENC/90S-12

AN INTERACTIVE SYSTEM OF COMPUTER GENERATED GRAPHIC DISPLAYS FOR MOTIVATING MEANINGFUL LEARNING OF MATRIX OPERATIONS AND CONCEPTS OF MATRIX ALGEBRA

THESIS

Presented to the Faculty of the School of Systems and
Logistics of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science In Systems Management

Stone W. Hansard, B.S., B.S.A.E.
Captain, USAF

September 1990

Preface

The purpose of this study was to determine whether reversing present learning strategies through a system of computer programs and simple procedures designed to increase the involvement of the student and teacher in the learning process would result in a more meaningful learning experience.

As a student of mathematics I have always preferred seeing an example of the mathematical concept or procedure before any equations were written on the chalkboard. It has also been my experience that once a professor has finished presenting a lecture or mathematical proof, the first question asked by students is "Can you show us an example?" The motivation for conducting this study was based on my perception of how students of mathematics prefer to learn, and a desire to make the learning of mathematics understandable, more enjoyable, and less tedious for the student.

My association with Dan Reynolds has transformed my general perceptions into a study which I hope will motivate teachers and students to seek more effective teaching techniques. Professor Reynolds' knowledge of learning theory, enthusiasm for educating, and desire to improve the quality of the educational experience has been invaluable in this effort to help my fellow students.

Stone W. Hansard

Table of Contents

List of Figures

AFIT/GSM/ENC/90S-12

Abstract

This ~~study~~ evaluated students' responses to a teaching method designed to involve students and teachers of mathematics in a meaningful learning experience using a system that allowed the student to participate in discovery learning. The system was comprised of a computer program that motivated visualization of concepts of matrix algebra, and a procedure that encouraged the students to verbalize each concept, create an algorithm for the matrix operation, and discover a mathematical formula that described the process for each operation. The teaching system is based on modern learning theory and reflects several recommendations made by the Commission on Standards for School Mathematics to the National Council of Teachers of Mathematics (NCTM). Of the 40 students who participated in the survey, over 87% felt the matrix program was more interesting than an ordinary lecture, 80% felt the system was more fun than reading a textbook, and 92% indicated they would use a learning tool similar to the matrix computer program if given the opportunity. Over 82% of the students believed the matrix teaching system helped them learn matrix algebra, while 72% felt the teaching system helped them learn matrix algebra better than they would have under typical classroom conditions.

AN INTERACTIVE SYSTEM OF COMPUTER GENERATED GRAPHIC DISPLAYS FOR MOTIVATING MEANINGFUL LEARNING OF MATRIX OPERATIONS AND CONCEPTS OF MATRIX ALGEBRA

I. Introduction

General Issue

A common belief held by students as well as educators is that the educator is a repository of concepts, and learning is the act of pouring those concepts into the waiting vessel that is the student. When this belief is put into practice, the result is a passive student whose sole responsibility is the memorization of concepts, a student that has no active involvement or experience in the learning process.

There exists a theory of learning proposed by Joseph D. Novak and D. Bob Gowin based on the works of David Ausubel, Joseph Schwab, and John Dewey that rejects the view of learning described above. Novak and Gowin perceive learning as a "change in the meaning of experience" and assert new knowledge cannot be constructed without the the active involvement of the student in the learning experience. They ask the question "How can we help individuals to reflect upon their experience and to construct new, more powerful meanings(1:xi)?"

One tenet of the theory contends that the question just posed, or any question concerning learning, cannot be adequately addressed "unless we consider simultaneously questions dealing with. . . teachers and how they teach, the structure of the knowledge that shapes the curriculum and how it is produced, and the social matrix, or governance of the educational setting(1:xii)."

At the Air Force Institute of Technology (AFIT) the governance is particularly rigid and the general curriculum only slightly less so. Given these restrictions, and drawing upon the theories of Novak and Gowin, the question can be asked: How can we help AFIT students "to reflect upon their experience and to construct new more powerful meanings"?

Specific Problem

For many students the study of mathematics is an onerous task; the concepts can be very involved and abstract. Mathematics is often taught by rote and the student takes little responsibility for his learning. Ausubel asserts that for meaningful, constructive learning to take place, the student must participate in the learning process(1:7). The problem becomes involving the student in the learning of mathematics in such a way as to facilitate meaningful learning which will cause knowledge construction, rather than memorization, and increase the student's grasp of mathematical concepts.

The Commission on Standards for School Mathematics released a report in 1989 which suggested that "Traditional teaching emphases on practice in manipulating expressions and practicing algorithms as a precursor to solving problems ignore the fact that knowledge often emerges from the problems(2:9)." A typical teaching scenario begins with the instructor providing equations to the students and demonstrating by example how the equations can be manipulated in order to solve problems. The Commission states further that traditional teaching techniques may need to be reversed(2:9). In this scenario the problem or example is given first and the student is allowed to discover the algorithm and the related mathematical expressions. Reversing the current teaching strategies may indeed result in greater student involvement and deeper learning.

Research Objective

The objective of this study is to determine whether reversing present learning strategies through a system of computer programs and simple procedures designed to increase the involvement of the student and teacher in the learning process motivates a more meaningful learning experience.

Research Hypotheses

1. Students given an active, as opposed to passive role in learning situations are more motivated and have a more meaningful learning experience.

2. A procedure that increases student-teacher interaction (such as the VVAM heuristic) motivates discovery learning.

3. The degree to which meaningful learning is achieved is dependent upon the temperament the student brings to the learning situation.

4. Students prefer to learn by reversing the process, i.e., working backward from an example of the mathematical process to arrive at the mathematical formula.

Scope of Research

The research subjects were restricted to Graduate Systems Management (GSM) students from the AFIT class GSM90S, and GSM and GCA students from the AFIT classes GSM91S and GCA91S. These students were required to take the Math Review for Engineering Managers (MATH 291). Only concepts presented in this course were selected because, according to David Ausubel, "The most important single factor influencing learning is what the learner already knows. Ascertain this and teach him accordingly"(3:vi). From MATH 291 concepts of linear algebra (herein called matrix algebra and matrix math) were chosen.

II. Background

Although there are many theories of learning, the theory proposed by Novak and Gowin was chosen for this study for two main reasons. First, the theory emphasizes meaningful learning as opposed to rote learning (heavily influenced by Ausubel) and second, because the theory recognizes that "an educational experience is a complex event"(1:6).

In their book "Learning How to Learn" Novak and Gowin state:

> To learn meaningfully, individuals must choose to relate new knowledge to relevant concepts and propositions they already know. In rote learning, on the other hand, new knowledge may be acquired simply by verbatim memorization and arbitrarily incorporated into a persons knowledge structure without interacting with what is already there. (1:7)

This is not to say there is no place for rote learning, but rather that there is a continuum with rote and meaningful learning occupying different positions on the continuum. Novak and Gowin also identify another continuum that flows ". . . from reception learning, where information is provided directly to the learner, to autonomous discovery learning, where the learner identifies and selects the information to be learned(1:7). Figure 1 shows how the two continua form a matrix and where some representative activities fit in the matrix(1:8). Attributes that Novak and Gowin associate with meaningful and rote learning are given in Figure 2.

MEANINGFUL LEARNING	Clarification of relationships between concepts	Well designed audio-tutorial instruction	Scientific research
	Lectures or most textbook presentations	School laboratory work	Most routine research
ROTE LEARNING	Multiplication tables	Applying formulas to solve problems	Trial and error
	RECEPTION LEARNING	GUIDED DISCOVERY LEARNING	AUTONOMOUS DISCOVERY LEARNING

Figure 1. Representative Learning Activities(1:8)

MEANINGFUL LEARNING	Non-arbitrary, non-verbatim, substantive incorporation of new knowledge into cognitive structure. Deliberate effort to link new knowledge with higher order, more inclusive concepts in cognitive structure. Learning related to experiences with events or objects. Effective commitment to relate new knowledge to prior learning.
ROTE LEARNING	Arbitrary, verbatim, non-substantive incorporation of new knowledge into cognitive structure. No effort to integrate new knowledge with existing concepts in cognitive structure. Learning not related to experiences with events or objects. No effective commitment to relate new knowledge to prior learning.

Figure 2. Attributes of Meaningful and Rote Learning(1:167)

A key word in the definition of meaningful learning is interaction, or more precisely, knowledge interaction. Knowledge interaction means that new knowledge interacts with knowledge a person already has.

An example of knowledge interaction and its relation to meaningful learning is evident in an ongoing project conducted by Thomas Banchoff at Brown University. In addition to the usual text book lectures on calculus, a weekly computer laboratory was instituted featuring an interactive program called VECTOR, which facilitates the graphing of curves and surfaces. Students take what they have learned in the classroom and use VECTOR to visualize curves and surfaces, and more importantly, they modify the program itself to fit their needs. When asked if there aren't better programs available "so that students and teachers won't have to go through the trouble of devising their own," Thomas Banchoff states, "No. This project has helped students in at least two ways: they have learned a tremendous amount about the mathematics involved and they have seen much more clearly the interrelationships of mathematics and computer science"(4:6). This example illustrates Novak's contention that, "As new experience is acquired and new knowledge is related to concepts already in a person's mind, these concepts become elaborated or altered and hence they can be related to a wider array of new information in subsequent learning"(5:25).

One common definition of learning states: "Learning [is] a relatively enduring change in behavior caused by experience"(6:115). Many theories of learning fail to address completely this concept of "experience". Joseph Schwab recognized the complexity in an educational experience, and in 1973, described 4 factors that are not ". . . reducible to any other, and . . . must be considered in educating"(1:6). The 4 factors, which Schwab termed commonplaces, consist of the teacher, the learner, the curriculum, and the milieu. Novak and Gowin describe the roles and responsibilities of each of the commonplaces:

> It is the *teacher's* obligation to set the agenda and to decide what knowledge might be considered and in what sequence [although a] skilled teacher will of course involve the learner in some aspects of the agenda planning. . . . The *learner* must choose to learn; learning is a responsibility that cannot be shared. The *curriculum* comprises the knowledge, skills, and values of the educative experience that meet criteria of excellence that make them worthy of study. The *milieu* [termed *governance* by Gowin] is the context in which the learning experience takes place, and it influences how teacher and student come to share the meaning of the curriculum. (1:6)

The governance and curriculum at AFIT are possibly the most rigid of any comparable institution with the exception of the service academies. Students do not have the luxury of determining the number of classes they take during a quarter, nor do they have a large or varied selection for the few electives they are allowed. Since there is little likelihood of affecting a positive change in the educational experience through governance or curriculum, the opportunity for improvement falls to the teacher and learner.

The theories of learning espoused by Ausubel, Schwab, Novak and Gowin, while not explicitly formulated for the learning of mathematics, have fundamental tenets which appear to be surfacing in response to the mathematics educating community's search for a workable set of standards for school mathematics.

In 1989 the National Council of Teachers of Mathematics (NCTM) established the Commission on Standards for School Mathematics for the express purpose of "[responding] to the call for reform in the teaching and learning of mathematics"(2:1). The Commission was charged with two tasks:

> 1. Create a coherent vision of what it means to be mathematically literate both in a world that relies on calculators and computers to carry out mathematical procedures and in a world where mathematics is rapidly growing and is extensively being applied in diverse fields.
>
> 2. Create a set of standards to guide the revision of the school mathematics curriculum and its associated evaluation toward this vision. (2:1)

In order to emphasize the importance of mathematical literacy, the "standards articulate five general goals for all students: (1) that they learn to value mathematics, (2) that they become confident in their ability to do mathematics, (3) that they become mathematical problem solvers, (4) that they learn to mathematical problem solvers, (4) that they learn to communicate mathematically, and (5) that they learn to reason mathematically"(2:5). Underlying these goals is the "premise that what a student learns depends to a great degree on *how* he or she has learned it"(2:5).

It is the "how" in this premise that will determine whether the student achieves these goals.

> In many class rooms, learning is conceived of as a process in which students passively absorb information, storing it in easily retrievable fragments as a result of repeated practice and reinforcement. Research findings from psychology indicate that learning does not occur by passive absorption alone(Resnick 1987). Instead, in many situations individuals approach a new task with prior knowledge, assimilate new information, and construct their own meanings. (2:10)

This passage from the Standards restates almost exactly certain tenets of learning theory expressed by Ausubel, Novak, and Gowin which can be found in the two preceding sections of this chapter.

It is unlikely that rote learning will result in the attainment of all five goals, and in recognition of that observation, the Standards state that ". . . 'knowing' mathematics is 'doing' mathematics. A person gathers, discovers or creates knowledge in the course of some activity having a purpose. This active process is different from mastering concepts and procedures"(2:7). This statement is undeniably Ausubelian in nature. The Standards go on to say that ". . . instruction should persistently emphasize 'doing' rather than 'knowing' [mathematics]"(2:7).

Again, this statement reflects the Ausubelian contention that the learning arises from the process not the product.

Ausubel stated that "The most important single factor influencing learning is what the learner already knows."(3:vi). The Standards take this observation into account: "We recognize that students exhibit different talents, abilities, needs, and interests in relationship to mathematics"(2:9). The Standards were not created to be a rigid set of guidelines; a rigid set of guidelines cannot take into account the differing backgrounds of students. Inflexible standards leave no room for the learning preference of the student. Instead, the Standards were proposed for three reasons: "(1) to ensure quality, (2) to indicate goals, and (3) to promote change"(2:2).

Although NCTM believes all three reasons are of equal importance(2:2), promoting change may be the most difficult to accomplish, particularly when the change is radically different from the status quo.

> Traditional teaching emphases on practice in manipulating expressions and practicing algorithms as a precursor to solving problems ignore the fact that knowledge often emerges from the problems. This suggests that instead of the expectation that skill in computation should precede word problems, experience with problems helps develop the ability to compute. *Thus, present strategies for teaching may need to be reversed; knowledge often should emerge from experience with problems* [italics the author's]. In this way, students may recognize the need to apply a particular concept or procedure and have a strong conceptual basis for reconstructing their knowledge at a later time. (2:9,10)

This thesis draws heavily upon the theories of learning expressed in the earlier sections of this chapter, and agrees with the premise that the "present strategies for teaching may need to be reversed." The remainder of this thesis documents the researcher's efforts to reverse present teaching strategies and increase student and teacher involvement in order to create a more meaningful learning experience for the student.

III. Methodology

Overview

The general method for solving this research problem was through a quasiexperiment in which 40 students from the AFIT classes GSM91S and GCA91S used a computer program and a suggested protocol designed to motivate the learning of concepts taken from matrix mathematics. The students responded to questions constructed to measure the extent to which they believed the visual and interactive aspects of the program and the use of the protocol did or did not motivate them to understand the concepts.

Several steps were completed prior to the actual experiment. The first step included selecting appropriate concepts from matrix math. Since computer graphics were to be used extensively, concepts that lent themselves to visual display were the most likely candidates. A number of concepts from linear algebra were selected from the AFIT course MATH 291.

The second step involved writing the computer program that visualized, and in some cases, animated the selected concepts. The program was also written so as to require the student to become involved in the learning process. The program was written in separate modules (a separate module for each concept), and each module was debugged and checked for accuracy.

The third step was the creation of a procedure or protocol that would require the student to become completely involved in the learning process. The protocol was also designed to guarantee student-teacher interaction.

The fourth step was the construction of the question set. The questions were designed to evaluate whether the system of computer program and protocol helped them understand the concepts, and if so, what aspects of the system were responsible. A Likert scale was used to measure the participants' responses.

In the actual experiment the students used the program and protocol and responded to the questionnaire. Since the GSM91S and GCA91S students attended lectures on the subject of matrix math in MATH 291, they were exposed to some extent to matrix math concepts prior to the experiment. This exposure before the experiment was not necessary because the program and protocol were designed to require no existing knowledge of matrix math. The questions about the experiment were not designed to measure how well the student had actually mastered the concepts, but whether or not the student felt that the program and protocol helped in motivating the learning and understanding of the concepts.

Concept Selection

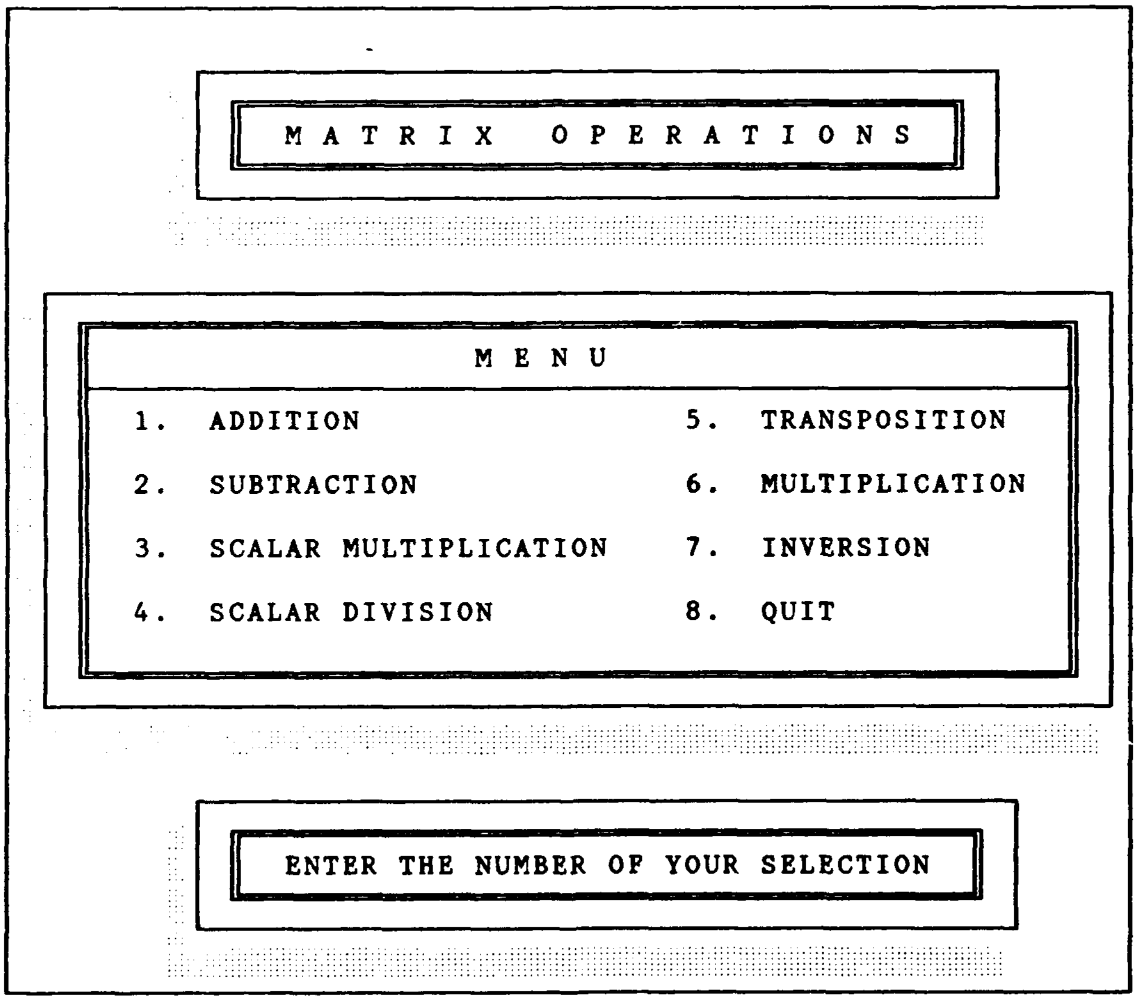
Although a wealth of mathematical concepts exist which would lend themselves to the programming and procedural treatment to be discussed later, the number of concepts under consideration was considerably narrowed by two constraints. The first constraint reflected the Ausubelian notion that the teacher should find out what the student already knows before beginning the learning process. This might have been accomplished by surveying the students with regard to their mathematics background, but since all of the students were to be exposed to the same concepts during the AFIT math review, it was decided to choose only from among those concepts presented during the review. All students would have at least the minimum concept base provided by the math review. The second constraint, which further narrowed the available concepts, arose from the timing of the AFIT math review and the time available for data collection and reduction. The earlier a concept was presented in the math review, the more time would be left to organize the data and reflect on the results of the study. The third constraint reflected the desire to share something with the students that would be of value during AFIT's required statistics sequence. Fortunately, written and verbal critiques of the previous year's math review and statistics sequence indicated a specific area of concern for many students. In response to those concerns, concepts from matrix mathematics were finally chosen.

Operations from matrix mathematics fulfilled the requirement that a concept be readily visualized using computer generated graphics; and a fortuitous consequence of selecting matrix operations was that the operations form a family of concepts that could be ordered from simple to complex. It was felt that the student would gain practice and confidence with the simpler concepts and graduate to the more complex.

The Computer Program

Having selected matrix operations, the next task required the actual creation of the computer program. There were at least three major influences on the final program configuration. First, some characteristics of the program logically followed from the choice of matrix operations. Second, other characteristics reflected certain principles of learning theory. Finally, the remaining characteristics resulted from limitations of the programming language (Microsoft QuickBASIC) and the researcher's background as a student and experience as a programmer. The influences were in no way mutually exclusive, and the overall characteristics of the program resulted from a combination of these influences.

The logical layout of matrix operations and the researcher's programming experience suggested a menu driven system with menu selections arranged from simple to complex matrix operations. The menu would allow the user to select a particular matrix operation at any time. Figure 3 approximates the main menu screen.



M A T R I X   O P E R A T I O N S

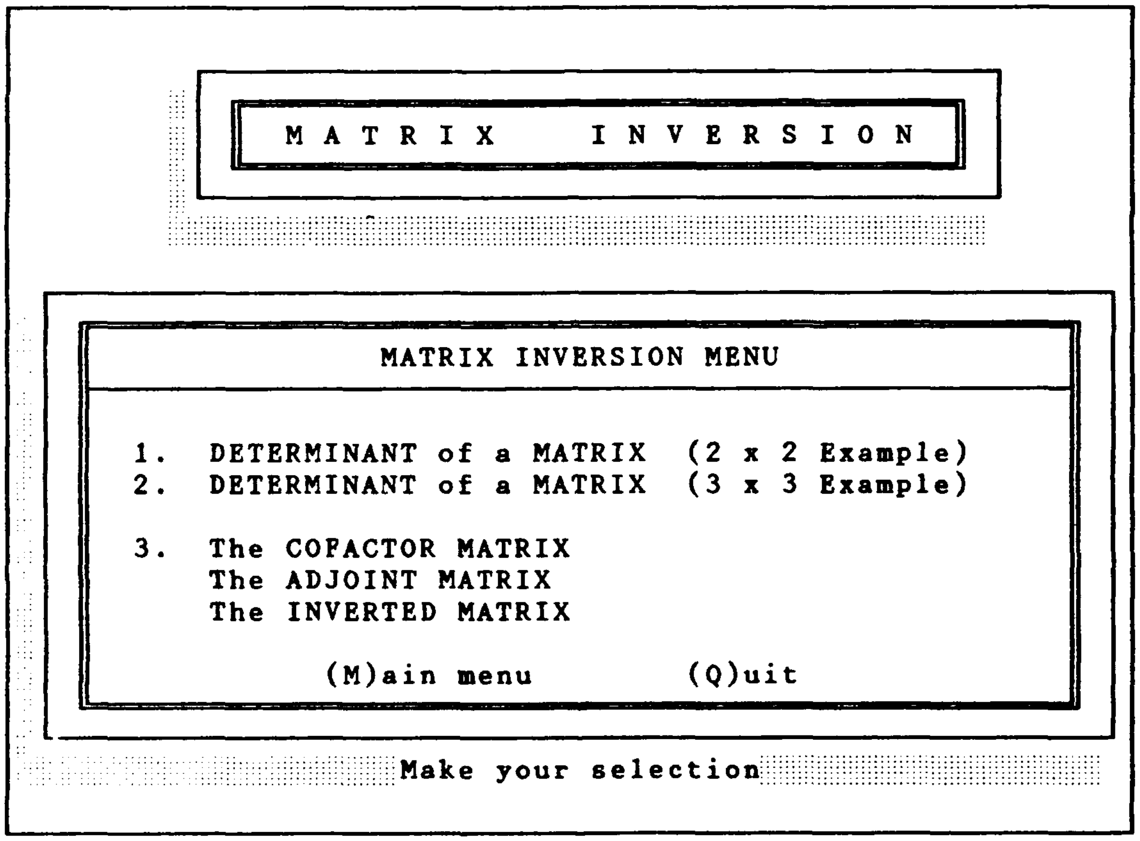
M E N U

1.	ADDITION	5.	TRANSPOSITION
2.	SUBTRACTION	6.	MULTIPLICATION
3.	SCALAR MULTIPLICATION	7.	INVERSION
4.	SCALAR DIVISION	8.	QUIT

ENTER THE NUMBER OF YOUR SELECTION

Figure 3. Main Menu Screen

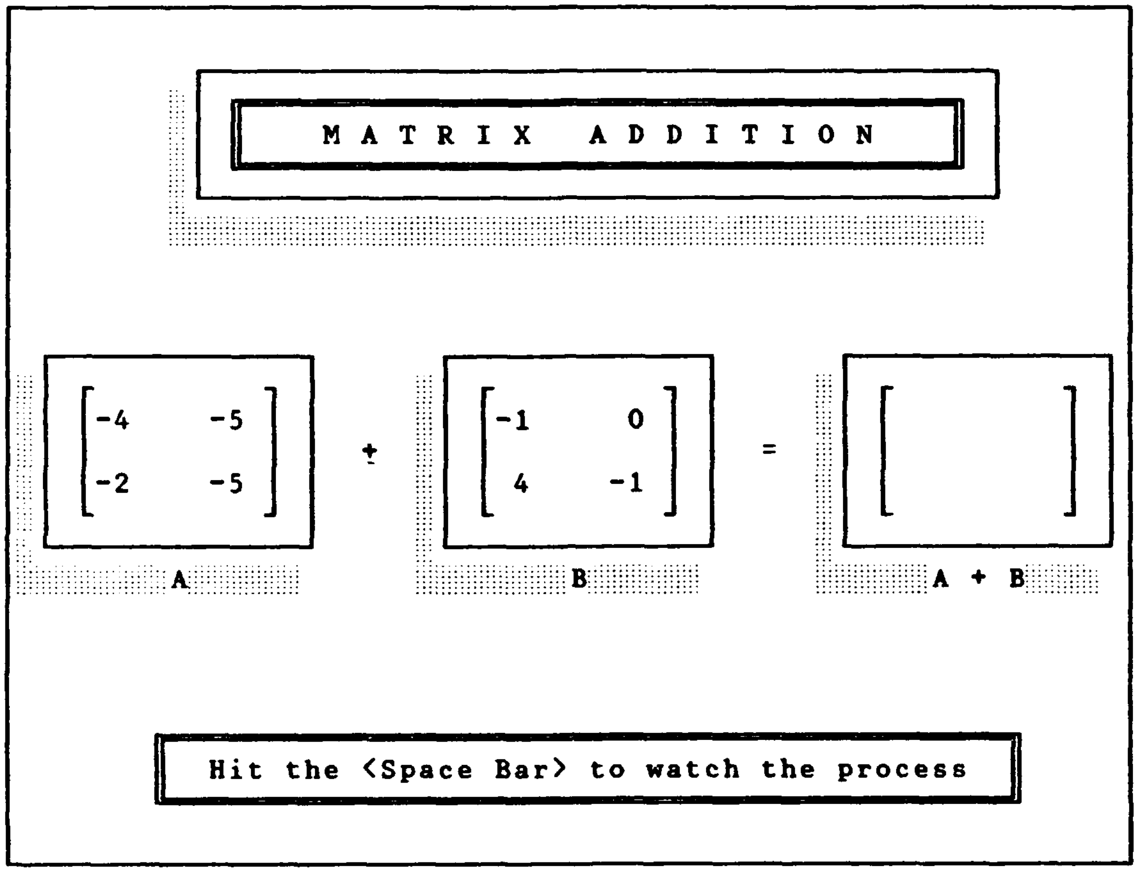
Since the last matrix operation, inversion, was the most complex and required several steps, selection of the inversion operation led to a secondary menu shown in Figure 4.



```
M A T R I X    I N V E R S I O N

MATRIX INVERSION MENU

1.  DETERMINANT of a MATRIX  (2 x 2 Example)
2.  DETERMINANT of a MATRIX  (3 x 3 Example)

3.  The COFACTOR MATRIX
    The ADJOINT MATRIX
    The INVERTED MATRIX

        (M)ain menu        (Q)uit

Make your selection
```

Figure 4. Inversion Menu Screen

Different matrix operations required different visualization formats; however, most of the process screens resemble the screen depicted in Figure 5.



$$\begin{bmatrix} -4 & -5 \\ -2 & -5 \end{bmatrix} + \begin{bmatrix} -1 & 0 \\ 4 & -1 \end{bmatrix} = \begin{bmatrix} & \\ & \end{bmatrix}$$

Hit the <Space Bar> to watch the process

Figure 5. Typical Process Screen

Although a few terms from matrix algebra were defined by screen comments where appropriate, in order to facilitate discovery learning, explanatory screen comments related to the conceptual aspect of matrix operations were kept to a minimum. For example, if a student were to try to multiply matrices with nonconforming dimensions, he or she would be informed that an error had occurred and would be invited to try again. The student would be allowed to discover the principle of conformability on his/her own if possible, or with help from the instructor if necessary. At this point it should be made clear that the computer program was in no way intended to be a stand-alone tutorial such as many Computer Assisted Instruction (CAI) packages. In keeping with the idea that learning is a complex event that takes place in the context of the four commonplaces described by Schwab(1:6), the program was designed to evoke student-teacher interaction rather than remove the teacher from the learning process.

In order not to detract from the learning process and to allow the student to concentrate on the chosen concept, the program was designed to require a minimum number of user inputs and keystrokes. Most menu selections require only the matrix dimension(s); some menu selections, such as matrix transposition, require an additional input concerning the selection of matrix rows or columns. Once dimensions are entered, the program randomly generates the element values (and scalar values if necessary). Since the concept to be demonstrated is independent of the matrix values, the student is relieved from the tedious task of selecting and typing in the values for a given matrix. In the case where selection of element values has an impact on the concept in question, such as selecting elements in order to create a singular matrix, the program assigns a probability of occurrence to the event so that the event will happen with some regularity. For example, a three by three matrix that is randomly generated for the matrix determinant selection will be singular approximately one time out of every five selections.

Once the matrix operation has been selected and the element values generated, the only action required of the student is: Hit the <Space Bar> to watch the process. The student may step through the process at any speed he or she wishes. When the process for a particular matrix operation is finished the student is given the choice of repeating the same process, returning to the previous menu, continuing to the next operation (if it is part of a sequence of operations), or quitting the the program. Some operations, such as calculation of the determinant, allow the student to use the same matrix elements again and again; this is to allow the student to discover that the same determinant will result regardless of which row or column is selected for matrix expansion. Other matrix operations generate a new set of elements every time the operation selection is made allowing the student to practice the operation with a variety of element values and avoid incorrect conclusions about the process that might occur due to confusing configurations of elements. For example, the student may not be able to correctly discover the process of matrix transposition if all of the matrix elements are the same. It is highly unlikely that repeating the operation would create the same matrix when the numbers are randomly generated.

The maximum matrix size was determined by the text screen format (80 columns and 25 rows) available on the Zenith 248 microcomputers at AFIT. The screen format allowed matrix dimensions of no more than three rows by three columns in order to accommodate the display of three matrices simultaneously. The program was tested on a number of different computer systems. Technical information is included in Appendix A and the source code in Appendix B.

The program makes extensive use of colors, blinking prompts, and some animation. Although an extensive organon exists about the visual aspects of information display, this study relied solely on the researcher's personal preferences for screen displays and valuable suggestions from outside sources.

The VVAM Protocol

Since the computer program was not designed as a stand-alone package, a procedure was developed to facilitate student-teacher interaction. This procedure consists of four main steps: Visualization, Verbalization, Algorithmization, and Mathematization (VVAM). Herein, the procedure is called the Protocol or VVAM Protocol.

Visualization. In the first step, the student selects the desired matrix operation and carefully observes the operation in action. The student may repeat the process as often as necessary and as slowly or quickly as desired.

Depending upon which operation has been selected, the student may vary certain parameters, such as matrix dimensions, in order to see if the observed process is consistent through a range of matrix sizes. The computer program dominates this step. The teacher can observe the process with the student and encourage him or her to look for patterns in the process or to experiment with different matrix dimensions; however, the primary interaction is between the student and the computer. Figure 6 shows graphically the changing role of the computer as each step of the VVAM protocol is encountered.

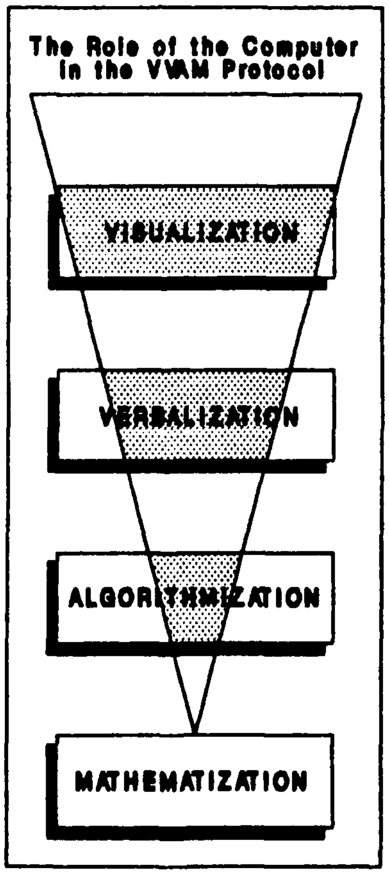


Figure 6. The Role of the Computer in the VVAM Protocol

Verbalization. In this step the student is asked to state in plain English the process just observed. The suggested technique is to have the teacher or another student listen to the verbalization and try to follow the process. If the student does not describe the operation precisely, or if a step in the process is misplaced or left out, the teacher can prompt the student to be more exact or suggest that the student visualize the process or part of the process again. Some students may be able to verbalize a matrix operation without returning to the visualization step. Others students may wish to verbalize as they observe the operation in action on the computer. At this point it is unnecessary for the student to use terms from matrix algebra in the verbalization; it is only important that the student display his understanding of the process by accurately describing the process to the teacher or a partner.

Algorithmization. The third step of the VVAM protocol requires the student to create a rigorous algorithm for the matrix operation under study. Although knowledge of a programming language would be helpful, only a a basic understanding of the structure behind an algorithm or computer program is needed. If necessary the teacher can introduce the student to data structures, input/output, sequencing of instructions, branching (conditional and unconditional), and iteration (looping). The student is asked to write down the algorithm on a piece of paper or chalkboard using any written sequence of statements that fully communicates the algorithm to the teacher (pseudocode). Once again, should the algorithm fail to precisely describe the matrix operation, the student may be asked to verbalize the process in order to discover where the error occurs. The student may also return to the computer to watch the process. An excellent way to verify the accuracy of the algorithm is by stepping through the algorithm and the program at the same time to see if both yield the same results at every step.

Mathematization. The final step of the VVAM protocol is the most rigorous of all. This step requires the student to write down the actual mathematical formula in algebraic form. The student must have an understanding of basic algebraic operators, variables, constants, and the more complex operators such as summation (sigma) and multiplication (pi). Since each step of the protocol must be successfully completed before continuing to the next step, the student should not have to go back any further than the algorithm in order to write down a formula for the matrix process. The teacher may at this time introduce the student to matrix notation.

Experimental Design Justification and Validity

The 40 students who took part in the experiment could be considered a census of the classes GSM91S and GCA91S, or a sample of all students who will attend AFIT in those two degree plans. If any generalization is to be made, then the students should be considered a sample, and some assumptions should be made to be reasonably confident that any threats to external validity are recognized.

The choice of population to which the researcher wishes to generalize his experimental results determines to a great extent the external validity of those results; however, a case can be made for logically generalizing results to a population even if it appears that the sample is not representative of that population.

If the population is defined as all future GSM and GCA students, then there should be assurances that the future students are indeed part of the same population. The rather inflexible entrance requirements imposed by AFIT all but mandate that this be the case. Standards for undergraduate grade-point averages, minimum requirements for college level mathematics courses, and a minimum number of slots available to other than U.S. military personnel are just a few of the restrictions that support the assumption that any given class of GSM and GCA students is a sample from a single population.

If the population is defined as all graduate students attending a university with graduate entrance requirements similar to those required by AFIT, then further assumptions could be made (and tested) to allow a degree of confidence in the generalization of this sample to a wider population. For example: it could be argued that although the sample is almost entirely military, at the time each student was earning his/her undergraduate degree, he/she was a member of the same civilian undergraduate population.

It is apparent that as this sample is generalized to wider and wider populations, the external validity becomes more and more questionable. For the purpose of this experiment, only the most justifiable assumptions were made to generalize the results to a given population.

The experiment is, in one sense, a "separate sample pretest-post-test"(7:127). The group consists of the GSM91S and GCA91S students; the treatment is the exposure to the computer program and protocol designed to motivate learning; and the pretest and post-test are simulated by a single test (Likert survey) that measures whether the student believes his learning was motivated more, less, or not at all after the treatment.

Although the external validity of this experiment is not guaranteed, and in fact, depends on the population in question, steps were taken to reduce some likely threats to the experiment's internal validity. First, selection of the measurement questions determines one aspect of the internal validity of the experiment; this is one factor over which the researcher has direct control. The measurement questions were designed to elicit responses to specific parts of the experiment as well as responses to the experiment in general. A check on internal validity can be made by comparing the responses to both kinds of questions. If the response to each specific question is favorable, then one would expect the response to a general question, which reflects the sum of the specific questions, also to be favorable. A second threat to internal validity, "maturation"(7:117), was moderated by testing the students as soon as possible after the completion of the matrix math lectures and the actual experiment. Finally, all students were tested at the same time so that any maturation that might have occurred would be essentially the same for all students.

Potential threats to the internal validity of the experiment might arise from the sequence and length of the question set. Questions were not randomly ordered. Questions that fell into general categories, such a Visualization or Verbalization, tended to be grouped together. One possible result could be that the answer to one question might influence the response to the following question. A benefit to the question sequence, however, could be that the student did not have to mentally "change gears" for each question. Surveys often ask essentially the same question in a number of different ways in order to provide a check on responses. Because of time restrictions, most questions were asked only once and no attempt was made to word questions so that an equal number of positive (agree) and negative (disagree) responses occurred.

IV. Analysis, Results, and Conclusions

Pretesting

Several students from the class GSM90S participated in testing the Matrix Program and Protocol. The students used the same Matrix Program and Visualization, Verbalization, Algorithmization, and Mathematization procedure that was presented to the GSM91S students. The reactions and comments from the pretest group were to be used to modify the Matrix Program and Protocol; however, the responses from the students were almost entirely positive, so no significant changes were made to the Matrix Program or Protocol. The most common reaction to the System from the students of class GSM90S was: "I wish [this system] had been around when we were starting AFIT."

Although the program and protocol were not changed after the pretest, there are two reasons why the pretest and actual experiment are not identical. First, the GSM90S students had already completed the Math Review and had gone on to use matrix algebra in the following statistics courses; however, any extra practice accrued by the upper class seems to have been entirely moderated by the six months intervening the last statistics class and the pretest. Second, students from the upper class were given the pretest in a one-on-one situation while the lower class participated in the experiment as a group and not all students had the opportunity to participate in the one-on-one situation.

The Experiment

The formal presentation of the Matrix Program and Protocol took place over a two day period in June 1990. The GSM students, numbering 25, and the GCA students, numbering 15, attended regularly scheduled math lab classes on the first day, and volunteers from both classes, numbering 10, attended a second session on the following day.

The first two-hour session began with the GSM students. Because of the size of the class it was necessary for students to share the personal computers, two students to a computer. Every student was given a copy of the matrix program and a protocol handout (see Appendix C).

The session began with an introduction of the protocol. The students were given time to read the protocol handout while its more important aspects were identified verbally. Next the program was introduced. An animated introduction screen was used to identify the need for student-teacher interaction and to explain that the Matrix Program and Protocol were designed to do just that. The students were encouraged to browse through the menu system and "get a feel" for the program operation. When the students finished experimenting with the program, they were asked to return to the main menu and begin with the matrix addition selection.

Per the protocol, the students were told to take their time and to try and understand the process they were visualizing on the computer screen. They were also encouraged to start over whenever necessary and to experiment with matrices of different dimensions.

When the students felt they completely understood the process of matrix addition, a volunteer was asked to verbalize, in plain English, the process in such a way that the process could be correctly followed and understood by an instructor or another student. If the operation could not be followed, the error was usually due to imprecise verbalization by the student or the student's incorrect grasp of the process. In the former case, the instructor guided the student to acceptable verbalization by asking the student to explain the process in another way. In the latter case, the student was asked to make sure if the process did indeed occur as verbalized by using the program to visualize the process once again. Students were encouraged to use the computer as they verbalized so that they were confident that their verbalization reflected precisely what they were visualizing.

After successfully verbalizing the matrix addition process, the students were given a short review to help them with writing a pseudo-code algorithm. The students were reintroduced to data structures, input/output, sequencing of instructions, branching (conditional and unconditional), and iteration (looping). A volunteer was asked to use the blackboard and carefully write an algorithm which correctly described the matrix addition process. Since the algorithm could be written in any combination of English and/or a programming language known to the student, the only notational requirements were that the algorithm could be explained and understood. As with the verbalization step of the protocol, errors either occurred because of the precision required in constructing the algorithm, or because the process was not fully understood by the student. Again, the instructor guided the student through the algorithmization by asking him to double check the steps of his algorithm, or if the student had incorrectly comprehended the process, he was invited to return to the program to find the flaw in his understanding of the matrix addition operation. When the student believed the algorithm to be correct, he was able to check each step in his algorithm with the corresponding step performed by the computer program.

In the final step of the protocol, the student volunteer was asked to write the mathematical equation(s) that would represent the algorithm. Since the students had been introduced to the mathematics package, MathCAD, they were allowed to use MathCAD notation if they preferred. The students were reminded that all algebraic operators were at their disposal. No students had to go back any further than the algorithm to successfully write the mathematical formula for matrix addition.

The second two hour session was performed in the same manner with the GCA students, and a final five hour session was performed with volunteers from both classes on the following day. The five hour session required no introduction and the students chose which matrix operations they wished to learn using the program and protocol. The students were encouraged to experiment with the program and protocol for a week and told that they would be asked to answer approximately 35 survey questions during the next scheduled math lab.

Description of the Survey

The survey questions were written to elicit responses in three main areas. The first area dealt with the actual layout and functionality of the computer program. Since the intent was to design a program that facilitates rather than intrudes on the learning process, questions were asked concerning the program's simplicity, and functionality. The second and largest area of interest was the VVAM protocol. Questions in this area were directed toward the four main steps: Visualization, Verbalization, Algorithmization, and Mathematization; and the two way relations between steps. These questions were intended to determine the students' feelings about the usefulness and/or necessity of each step in the procedure. The third and most general area of interest was concerned with the students' overall reaction to the learning experience. Some questions were asked so that the student could compare the Matrix Program and VVAM Protocol with traditional learning experiences such as classroom lecture and textbook reading; other questions dealt with the need for student-teacher interaction and the responsibilities of the student and teacher during the learning process.

A Likert scale format was chosen to present the survey questions. A computer program was written (see Appendix E) which allowed the students to select one of the following responses to any given question: Strongly Disagree, Disagree, Undecided, Agree, or Strongly Agree.

Statistical Test

Background. The Myers-Briggs Type Indicator (MBTI) is a research instrument that has application in learning theory. If a teacher can determine how a student prefers to learn, the teacher may be able to use an appropriate teaching technique. The MBTI attempts to identify a person's psychological type by ascertaining the following four preferences:

1. Does the person's interest flow mainly to the outer world of actions, objects, and persons [Extroversion]; or to the inner world of concepts and ideas [Introversion]?

2. Does the person prefer to perceive the immediate, real, practical fact of experience and life [Sensing]; or the possibilities, relationships and meanings of experience [Intuition]?

3. Does the person prefer to make judgments or decisions objectively, impersonally, considering causes of events & where decisions may lead [Thinking]; or subjectively and personally, weighing values of choices & how they matter to others [Feeling]?

4. Does the person prefer mostly to live in a decisive, planned and orderly way, aiming to regulate & control events [Judgment]; or in a spontaneous, flexible way aiming to understand life and adapt to it [Perception]? (8:13)

Although the survey questions were not written to address learning preference, it was felt that responses to certain questions might reveal a dependence on psychological type, particularly as related to the Introversion versus Extroversion preference and the Sensing versus Intuition preference. All students in the sample had taken the Indicator test as part of AFIT in-processing procedures and 33 of the 40 students provided their MBTI codes when answering the demographic questions in the survey.

Contingency Tables. A procedure from *STATISTIX: An Interactive Statistical Analysis Program for Microcomputers* was used to test whether responses to the survey questions were dependent on psychological type as determined by the MBTI. A two by two contingency table using a Chi Square test for independence was selected as the appropriate test.

For the purpose of this test, "Undecided" responses were omitted, "Agree" and "Strongly Agree" were combined in the category "Agree", and "Disagree" and "Strongly Disagree" were combined in the category "Disagree".

Each of the survey questions was tested against the students' MBTI for Introversion/Extroversion and Sensing/Intuition. Most of the questions were not expected to show dependencies on psychological type, but those questions dealing with Verbalization had the most potential for revealing dependencies on Introversion/Extroversion preference.

I prefer to THINK privately about what I know rather than try to objectively display my knowledge by verbalization.

	Disagree	Agree
Extrovert	4	3
Introvert	3	17

PEARSON'S P-value .0285

Figure 7. MBTI Related Responses

At a significance level of 95%, only one survey question resulted in responses that indicated dependence on psychological type as determined by the MBTI. Figure 7 shows the survey question and the results of the STATISTIX test. Note the P value is well below the .05 significance value.

Results of the Survey

The following figures display the survey data in the form of frequency histograms and report the descriptive statistics on which most conclusions were based. The questions and results shown in the figures represent the three main areas of interest described in the previous section. Appendix F contains every survey question with raw and statistical data in the same format as seen in the following figures.

To simplify the breakout of responses into percentages the five possible responses were reduced to three. Strongly Agree and Agree were combined in a single category named Agree, and Strongly Disagree and Disagree were combined in a single category named Disagree. The Undecided category was not changed.

Program Functionality. Figure 8 shows the survey question and results for a question taken from the first main area of interest, the functionality of the Matrix Program.

The mechanical simplicity of the Matrix Program allowed me to concentrate on LEARNING MATHEMATICAL CONCEPTS rather than on HOW TO USE THE PROGRAM proper.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	1
3.	Undecided	2
4.	Agree	21
5.	S: Agree	16

Disagree - 2.5%
Undecided - 5.0%
Agree - 92.5%

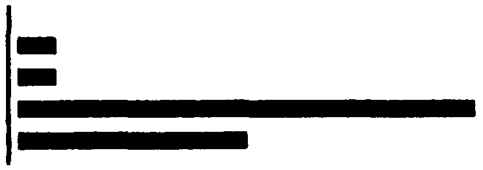
MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.300	.06869	4.000	2.000	5.000

Figure 8. Program Functionality Question

Three questions specific to the functionality of the Matrix Program were asked in order to determine if the goal of designing a program that motivated the learning experience without complicating the process had been achieved. The responses to all three questions indicated that at least 90% of the students found the program easy to use and helpful in discovering the essential aspects of matrix algebra. Another question, shown in Figure 9, combined aspects of functionality and the visualization process. Again, 90% of the students found the program very helpful in visualizing matrix operations in action.

The use of color graphics highlighted critical matrix operations and helped focus my attention on key activities of the process under study.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	2
4.	Agree	24
5.	S. Agree	12



Disagree	-	5%
Undecided	-	5%
Agree	-	90%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.150	.07355	4.000	2.000	5.000

Figure 9. Visualization Question

<u>The VVAM Protocol</u>. Since this was the most important area of interest and most closely tied to the research objective, the survey results concerning the VVAM protocol are discussed in terms of the original hypotheses. Each research hypothesis and a brief analysis of findings related to that hypothesis are presented below.

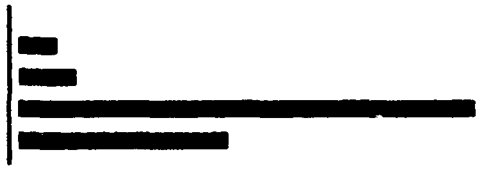
1. Students given an active, as opposed to passive role in learning situations are more motivated and have a more meaningful learning experience.

At least in the case of the Matrix Program and VVAM Protocol this hypothesis appears to be true. Students were able to participate actively in two ways.

First the student had to interact with the Matrix Program which was primarily concerned with the visualization step of the VVAM Protocol. Over 87% of the students felt that using the Matrix Program was more interesting than listening to an ordinary lecture (see Figure 10), while over 97% of those students with an opinion felt the Matrix Program was more fun than reading a text book. The assumptions here are that lectures and textbooks usually put the student into a passive learning role, and that the active role has a positive motivating influence on the student. If making a learning experience more interesting and fun is a positive motivating influence, then the Matrix Program has certainly succeeded.

Using the matrix program as a learning tool was more interesting than listening to an ordinary lecture.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	3
4.	Agree	24
5.	S. Agree	11



Disagree - 5.0%
Undecided - 7.5%
Agree - 87.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.100	7.442E-01	4.000	2.000	5.000

Figure 10. Active Use of the Matrix Program

Second, the student was encouraged to interact with the teacher; the amount of student-teacher interaction being primarily dependent upon the abilities of the student and the student's and teacher's predispositions toward introversion or extroversion. This student-teacher interaction is addressed in the next research hypothesis.

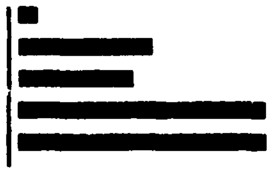
2. A procedure that increases student-teacher interaction (such as the VVAM heuristic) motivates discovery learning.

The overall student response showed that the Matrix Program and VVAM Protocol were preferred to typical classroom lecture and textbook learning techniques which generally consist of one way information transfer. In a one way transfer of information the student may truly believe he has grasped a particular concept when he has not. Even when a student asks questions of the instructor the new information is usually given, not discovered. The VVAM protocol ensures that a great deal, if not most of the communication, flows from the student to the instructor. The instructor's responsibility becomes one of learning guidance while the student is encouraged to discover the steps and sequence of a mathematical process. Every student who attempted the VVAM protocol made mistakes at some point. Students that were sure they understood a particular matrix operation often had to return to the visualization step to "discover" why their attempt at verbalization was not correct. Although survey questions can try to quantify the benefits of discovery learning, they cannot capture the positive reaction of the student when discovery learning takes place.

Two survey questions lend credence to the relationship between discovery learning and active participation. The first question shows that even though all the students surveyed were probably taught mathematics using typical teaching techniques, only 47% thought that it was the teacher's responsibility to ensure the student possessed an adequate concept base to learn any particular subject. This would indicate that about half of the students felt they were at least as responsible as the teacher for the learning experience. The second question indicated that 65% of the students felt that two way communication with an instructor was essential for a student to master the steps involved in performing any matrix operation (see Figure 11).

Two way communication with an instructor is essential for a student to master the steps involved in performing any matrix operation.

	RESPONSE	FREQUENCY
1.	S. Disagree	1
2.	Disagree	7
3.	Undecided	6
4.	Agree	13
5.	S. Agree	13



Disagree	- 20%
Undecided	- 15%
Agree	- 65%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.750	1.171	4.000	1.000	5.000

Figure 11. Student-Teacher Interaction

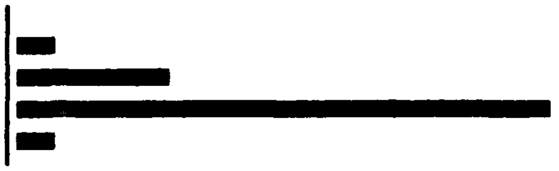
As was mentioned above, the simple matrix operations probably require less two way communication than the more complex operations. Because of the wording of the survey question the students' preference for two way communication is not entirely evident; however, the response does indicate that 65% of the students would prefer two way communication even for the simplest of matrix operations.

3. The degree to which meaningful learning is achieved is dependent upon the temperament the student brings to the learning situation.

This statement may seem more like a truism than an hypothesis; however, the results of the survey show that temperament as measured by the MBTI did not have as much influence as one might expect. As discussed earlier, only one survey question indicated any dependence on psychological type and the question simply reaffirmed that introverted students prefer to think privately rather than be asked to verbalize. The conclusion is that even though a student might have an MBTI suggesting some degree of introversion, the student still recognized the benefit to be gained from verbalizing a mathematical process and was willing to do so. This conclusion is based on survey results showing that although 55% of the students preferred not to verbalize, Figure 12 shows that only 5% believed verbalizing was not an excellent way to display one's knowledge about each matrix operation.

Verbalizing the steps in a matrix operation provided me with an excellent way to display my knowledge about each operation.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	8
4.	Agree	28
5.	S. Agree	2



Disagree - 5%
Undecided - 20%
Agree - 75%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.750	6.304E-01	4.000	2.000	5.000

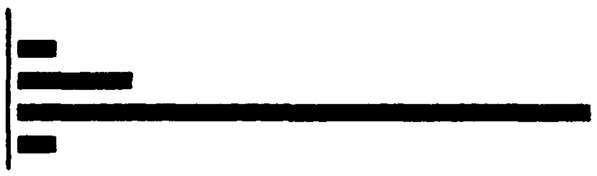
Figure 12. Verbalization Question

4. Students prefer to learn by reversing the process, i.e., working backward from an example of the mathematical process to arrive at the mathematical formula.

The Matrix Program and VVAM protocol were specifically designed to "reverse the process." Not every student was expected to be comfortable with each step in the procedure and it was expected that different students would bring different learning preferences to the experiment. Regardless of learning preference, 82% of the students (91% of those having an opinion) preferred to observe the process and visualize each step in the process before being given the mathematical formula. When asked if each step in the VVAM protocol facilitated the next step, 80% of the students thought visualization should occur before attempting to state the mathematical process rigorously; 72% of the students agreed that being asked to verbalize the process made it easier to write the algorithm; and over 87% felt that constructing an algorithm for a given matrix operation made the mathematical modeling of the process much easier. Finally, as shown in Figure 13, 80% of the students agreed that the order of the four steps in the VVAM protocol was appropriate. The lower percentage of students caring to verbalize before constructing an algorithm may result from the large number of students (23 of 33) whose MBTI indicated a higher degree of introversion. As stated in the discussion of the previous hypothesis, the interesting point is not the lower percentage of students caring to verbalize, but that the percentage is so high given the number of students who lean toward introversion.

The mathematical modeling of a matrix operation is facilitated by constructing an algorithm AFTER verbalizing the steps involved in the process under observation.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	6
4.	Agree	30
5.	S. Agree	2



Disagree	- 5%
Undecided	- 15%
Agree	- 80%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.800	6.076E-01	4.000	2.000	5.000

Figure 13. Sequence of Steps in the VVAM Protocol

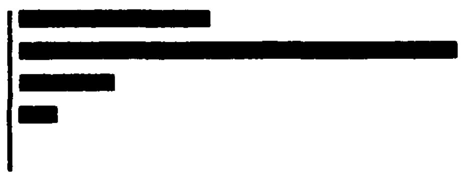
These responses show that one goal of the VVAM protocol, "reversing the process," was achieved. Whether or not reversing the process made any difference to the student is discussed in the next section.

Reaction to the Overall Learning Experience. Several survey questions were directed at determining the students' reaction to the learning experience as a whole. Although the questions specific to the functionality of the Matrix Program and the steps of the VVAM protocol yielded important information about the learning system, the general questions gave important information about the learning experience.

Since a learning experience is not worthwhile if the learning goal is not achieved, one question simply asked if the student thought the goal of learning about matrix algebra had been achieved. Since nearly 60% of the students had taken an undergraduate course in matrix algebra and over 87% had learned matrix algebra in conjunction with some other course, one might have expected several students to find the use of the Matrix System an unnecessary learning experience; this was not the case. As shown in Figure 14, only two of the 40 students felt that the Matrix System had not helped them learn about matrix algebra.

I don't think the HANSARD MATRIX SYSTEM helped me learn much about matrix algebra at all.

	RESPONSE	FREQUENCY
1.	S. Disagree	10
2.	Disagree	23
3.	Undecided	5
4.	Agree	2
5.	S. Agree	0



Disagree	- 82.5%
Undecided	- 12.5%
Agree	- 5.0%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
1.975	7.675E-01	2.000	1.000	4.000

Figure 14. Reaction to the Overall Learning System

One of the students who did not benefit from the exercise did indeed have a background in matrix algebra; however, answers to other survey questions indicate that his or her learning preference may have been more responsible for the unsatisfactory learning experience. Responses to several questions show that the student strongly preferred one way communication from the teacher to the student and that he/she believed the teacher to be solely responsible for providing mathematical concepts. The student did not want to visualize matrix operations and did not like any aspect of the Matrix Program. This was the only student to state he/she would not use a learning tool similar to the Matrix Program again. All of the student's learning preferences appear diametrically opposed to the learning heuristic the Matrix Program and VVAM Protocol were intended to motivate.

Unlike the first student, the second student had never been exposed to matrix algebra; also, this student was highly motivated by the Matrix Program and the VVAM Protocol, and strongly preferred two way communication with the teacher. This student also indicated that programs like the Matrix System should be constructed for all types of mathematical subjects and that he or she would use learning tools similar to the Matrix Program if given the opportunity. Finally, the student indicated that the VVAM Protocol helped him/her understand matrix operations better than would have been the case under typical classroom conditions. These responses indicate that the student may have entered an unintended response when answering the question about having learned m\ h about matrix algebra.

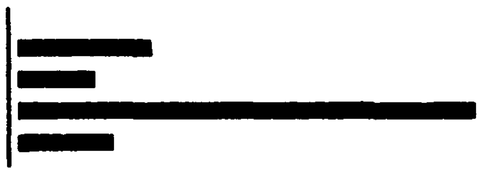
With at least 82% of the students stating the Matrix System helped them learn about matrix algebra, a case can be made that the Matrix System met its goal. When considering the mathematical background of the students; that case is even stronger.

If the Matrix System met its goal then the VVAM Protocol and Matrix Program must have achieved their separate but related goals. Survey questions discussed earlier addressed the steps of the Protocol and functionality of the Matrix Program; the following questions helped determine the students' general feeling toward both components of the Matrix System.

One question asked the students to make a comparison of the VVAM Protocol with previous learning experience. As shown in Figure 15, over 72% thought that the VVAM heuristic helped them understand matrix operations better than they could have under typical class room conditions.

I think the VVAM protocol helped me understand matrix operations better than I could have understood them under typical classroom conditions.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	7
3.	Undecided	4
4.	Agree	24
5.	S. Agree	5



Disagree	- 17.5%
Undecided	- 10.0%
Agree	- 72.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.675	9.167E-01	4.000	2.000	5.000

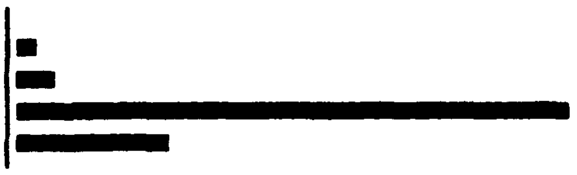
Figure 15. Reaction to the VVAM Protocol

This 72% is likely to be low for one important reason. Because of time constraints, most students saw only the matrix operation of addition performed. Since matrix addition is a very simple operation, many of the students might have felt that a typical classroom lecture was adequate. The 10 students who attended the optional presentation on the second day were introduced to the more complicated steps required to perform a matrix inversion. These students became absorbed in the VVAM heuristic and the "two hour" session lasted for five hours. Not many students find classroom lectures sufficiently motivating to want to attend one for five hours --- on a Saturday!

A different approach was used determine to the students' attitude toward the Matrix Program. As shown in Figure 16, over 92% of the students would use a similar learning tool again.

I would use learning tools similar to the MATRIX program if given the opportunity.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	1
3.	Undecided	2
4.	Agree	29
5.	S. Agree	8



Disagree	- 2.5%
Undecided	- 5.0%
Agree	- 92.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.100	5.905E-01	4.000	2.000	5.000

Figure 16. Reaction to the Matrix Program

The response to this question and the responses to questions concerning the functionality of the Matrix Program show the program to be motivational and a useful learning tool which enhances the learning experience without complicating the process.

The results of the survey show that the Matrix Program and VVAM Protocol successfully reversed what the Commission on Standards for School Mathematics termed "Traditional teaching emphases on practice in manipulating expressions and practicing algorithms. . . [which] ignore the fact that knowledge often emerges from the problems(10:9)." The Matrix System also encouraged active student participation, student-teacher interaction and discovery learning, the benefits of which are endorsed by learning theory and quantified by the students' response to this study. In the final analysis this researcher's desire to help his fellow students learn concepts from mathematics has been fulfilled.

V. Recommendations

This study began with a desire to help future AFIT students learn, as best they could, mathematical concepts required in their course work. The researcher's preference for seeing an example first and his background in computer programming suggested the use of computer generated graphics to display the example or mathematical process. Modern learning theory suggested the value of discovery learning, active participation, and student-teacher interaction. Since a simple computerized tutorial could not fulfill all of the requirements, a procedure was needed to tie the visualization of the mathematical concept to the final mathematical formula, and in doing so, ensure the student's active participation. Professor Dan Reynolds provided that procedure in the form of the VVAM Protocol which bridged the gap between visualization and mathematization by encouraging the student to verbalize the process and write an algorithm to describe the process.

Since the survey indicated that the students responded favorably to this new learning heuristic, further studies should be conducted to gain insight into this learning process; because this experiment was necessarily preliminary in nature, these further studies could take any of several paths.

One recommended path would deal mostly with the sample population. The original experiment was restricted to a small sample of postgraduate students. The Matrix Program and VVAM protocol could be left intact and presented to larger and more varied populations. Undergraduates and high-school students would certainly be capable of employing the full heuristic, and a slightly modified VVAM protocol might give even younger students the opportunity to participate.

If the VVAM protocol can be modified to fit one population, then it could be modified to fit others. As was noted in Chapter IV, learning preference plays a part in the learning experience. One would expect introverted students to be less comfortable with the verbalization step of the Protocol. A student who is uncomfortable speaking to a classroom of 30 students may have no problem in a group with three or four classmates. One possibility could be to modify the verbalization step so that introverted students would feel more at ease. A modification to one or more steps of the Protocol could address one or more learning preferences.

In order to assess a student's learning preference, a screening method, such as the MBTI, would be invaluable; however, to evaluate the impact of learning preference on students' response to the VVAM protocol would require a set of well constructed survey questions. The original survey questions were more concerned with the students' general perceptions of the Protocol. A set of survey questions could be constructed specifically to address learning preference and temperament using established survey questionnaire techniques which were lacking in the original question set.

A host of mathematical concepts from algebra and calculus, as well as their applications, are rarely taught without some kind of visualization. Basic differential calculus almost always begins with concepts such as tangents to a curve. Integral Calculus might begin with the concept of limits, but quickly moves into determining areas under a curve with some technique such as the Trapezoidal Method. Given the appropriate computer visualization and the VVAM Protocol, students may be able to discover the Fundamental Theorem of Calculus in the same way this study helped students to discover concepts and operations of matrix algebra.

Given the success of the unmodified VVAM protocol and the fact that the Matrix Program was intended to help prepare students for the AFIT statistics sequence, perhaps the most logical follow on to this study would be to investigate statistical concepts which lend themselves to visualization. One such concept is in the area of hypothesis testing. Not only are normal distributions readily visualized, but Type I and Type II errors are represented by areas under the normal curves which represent the null and alternate hypotheses. The student could be allowed to vary parameters such as mean, standard deviation, and confidence interval, and visualize the effects on Type I and Type II error by observing the changing areas which represent those errors.

Other distributions also could be easily visualized. The Gamma distribution, for example, has many interesting characteristics and applications. By manipulating Gamma's parameters, α and β, the student could discover at what point Gamma becomes an Exponential or Chi-square distribution. Further manipulation could reveal why the Gamma distribution can be used to approximate a normal distribution or vice versa.

Visualization of discrete distributions could be carried out just as easily. One possibility would be to have the computer generate discrete data in the form of a histogram and allow the student to manipulate the parameters of selected continuous distributions in order to pick a continuous distribution which best models the discrete data. Such a program would underscore the relationship between discrete and continuous distributions.

A number of concepts from statistics rely on heuristics; a case in point is the Central Limit Theorem (CLT). Instead of providing the student with a rule of thumb that says a sample size of 25 or 30 is sufficient to apply the CLT, the student could vary sample sizes from "unknown" computer generated distributions and see just how large a sample size is needed to generate a sample mean which is normally distributed. The student could watch the distribution of the sample mean change with increased sample size and discover that fewer samples are required if the "unknown" distribution is approximately normal, while a larger sample size may be required if the underlying distribution is decidedly non-normal.

Most of the preceding recommendations deal with fairly basic concepts from calculus and statistics. The real benefit of the VVAM protocol and the appropriate visualization program may lie in helping students learn even more complex mathematics. Topics from advanced statistics might include regression analysis or the projective approach to linear models. A more advanced programming language would facilitate three-dimensional visualization which more advanced topics, such as the projective approach to the linear model, require; however, there is still a lot of work to be done with concepts that need only two-dimensional graphics.

Many programming tools are available, as are programmers capable of creating the graphics needed for the first step in the VVAM Protocol. All that is left is to get to work and put some finished products into the hands of the students and teachers; they'll take care of the rest.

Appendix A: Technical Information

PROGRAMMING LANGUAGE/ENVIRONMENT: BASIC/QuickBASIC 4.5

COMPUTER: IBM/AT Compatible Computer
80286 Processor
80287 Math Coprocessor
VGA Graphics Card

EXECUTABLE FILES:

1) MATRIX.EXE - Displays introduction screens and runs the main program, MAT20.EXE.

2) MAT20.EXE - Main Program. May be run separately from MATRIX.EXE.

3) TQ.EXE - Thesis Question program. Computerized survey questionnaire.

NOTES:

*** Color Monitor Required ***

1) QuickBASIC code was written to use EGA color graphics.

2) 80287 Math Coprocessor is not required for operation.

3) VGA graphics card is not required for operation.

4) EGA graphics card allows full operation of MATRIX.EXE, MAT20.EXE, and TQ.EXE.

5) CGA graphics card allows full operation of the main program, MAT20.EXE, only.

COMPUTER SYSTEMS TESTED:

1) TANDY (Radio Shack), MS-DOS Compatible

2) ZENITH-248, MS-DOS Compatible

3) IBM/XT Compatibles (8086 processor)

4) IBM/AT Compatibles (80286 processor)

Appendix B: Matrix Program Source Code

```
DIM getarray(100)
DIM aray1%(100), aray2%(100), aray3%(100), aray4%(100)
DIM aray5%(100), aray6%(100), aray7%(100)
SCREEN 9
COLOR 15, 8
CLS
PRINT ""
PRINT "AN INTERACTIVE SYSTEM OF COMPUTER GENERATED GRAPHIC_
DISPLAYS"
PRINT ""
PRINT "FOR MOTIVATING THE MEANINGFUL LEARNING OF MATRIX_
 OPERATIONS"
PRINT ""
PRINT ""
PRINT ""
PRINT "                                         THESIS"
PRINT ""
LOCATE 25, 1
COLOR 14
PRINT ; "                        <Hit any key to continue>";
COLOR 15
VIEW PRINT 12 TO 24

PRINT "Presented to the Faculty of the School of Logistics"
PRINT ""
PRINT "of the Air Force Institute of Technology"
PRINT ""
PRINT "Air University"
PRINT ""
PRINT "In Partial Fulfillment of the"
PRINT ""
PRINT "Requirements for the Degree of "
PRINT ""
PRINT "Master of Science in Systems Management"
PRINT ""
DO
LOOP UNTIL INKEY$ <> ""
PRINT "Stone W. Hansard, B.S., B.S.A.E"
PRINT ""
PRINT "Captain, USAF"
PRINT ""
PRINT ""
PRINT ""
PRINT "September 1990"
PRINT ""
PRINT ""
PRINT ""
PRINT ""
PRINT "Approved for public release; distribution unlimited"
DO
LOOP UNTIL INKEY$ <> ""
VIEW PRINT
'==========================================================
' START INTRO SCREEN
'==========================================================
COLOR 15, 0
PI! = 3.1416
CLS
clx% = 250
cly% = 260

kk% = 1

LINE (190, 200)-(186, 202), 12
LINE (184, 200)-(190, 200), 12     'straight line
LINE (190, 200)-(186, 198), 12

LINE (200, 200)-(204, 202), 12
LINE (200, 200)-(206, 200), 12     'straight line
LINE (200, 200)-(204, 198), 12

GET (184, 198)-(206, 202), aray7%
LINE (184, 198)-(206, 202), 0, BF

'generate arrows

LINE (100, 100)-(112, 112), 11       'x to make up
LINE (100, 112)-(112, 100), 11       'and down arrows
LINE (101, 100)-(113, 112), 11       'x to make up
LINE (101, 112)-(113, 100), 11       'and down arrows
LINE (99, 100)-(111, 112), 11       'x to make up
LINE (99, 112)-(111, 100), 11       'and down arrows

GET (107, 102)-(112, 110), aray3%    'left arrow
'GET (100, 102)-(105, 110), aray4%    'right arrow

GET (99, 102)-(113, 105), aray2%    'down arrow
GET (99, 107)-(113, 110), aray1%    'up arrow

LINE (100, 100)-(112, 112), 14       'x to make up
LINE (100, 112)-(112, 100), 14       'and down arrows
LINE (101, 100)-(113, 112), 14       'x to make up
LINE (101, 112)-(113, 100), 14       'and down arrows
LINE (99, 100)-(111, 112), 14       'x to make up
LINE (99, 112)-(111, 100), 14       'and down arrows

GET (100, 102)-(105, 110), aray4%    'right arrow
GET (99, 102)-(113, 105), aray5%    'down arrow
GET (99, 107)-(113, 110), aray6%    'up arrow
```

```
LINE (99, 100)-(113, 112), 0, BF      'erase x

PRESET (clx% - 195, cly% - 15)    ' locate graphic pixel

CIRCLE STEP(30, 0), 20, 14, .3 * PI!, PI!
CIRCLE STEP(25, -10), 15, 14, .3 * PI!, .91 * PI!
CIRCLE STEP(30, -10), 20, 14, .2 * PI!, PI!
CIRCLE STEP(30, -10), 20, 14, 1.9 * PI!, .95 * PI!
CIRCLE STEP(39, 8), 20, 14, 1.9 * PI!, .9 * PI!
CIRCLE STEP(39, 8), 20, 14, 1.9 * PI!, .9 * PI!
CIRCLE STEP(20, 18), 20, 14, 3 * PI! / 2, PI! / 2
CIRCLE STEP(-9, 22), 20, 14, 3 * PI! / 2, PI! / 6
CIRCLE STEP(-43, -30), 76, 14, 2 * PI! / 1.41, 3.4 * PI. / 2
CIRCLE STEP(-41, 30), 40, 14, 2 * PI! / 1.5, 3.4 * PI! / 2
CIRCLE STEP(-33, -30), 76, 14, 2.4 * PI! / 1.9, 3.1 * PI!/2
CIRCLE STEP(-53, 20), 31, 14, PI! / 1.5, 3.1 * PI! / 2

COLOR 11
LOCATE 1, 18
PRINT "MEANINGFUL LEARNING"
LOCATE 3, 18
PRINT "    requires dialog,"
LOCATE 4, 18
PRINT "              exchange,"
LOCATE 5, 18
PRINT "               sharing, "
LOCATE 6, 18
PRINT "                and sometimes compromise."
COLOR 14
LOCATE 8, 14
PRINT "The AUTHORS of this software are happy to have the"
LOCATE 9, 14
PRINT "opportunity to LEARN about MATRIX ALGEBRA with you."
COLOR 15
LOCATE 15, 9
PRINT "STUDENT"
LOCATE 15, 65
PRINT "EDUCATOR"

CIRCLE (90, 180), 20, 15        ' STUDENT
CIRCLE (83, 176), 3, 7       ' STUDENT LEFT EYE
CIRCLE (97, 176), 3, 7       ' STUDENT RIGHT EYE
LINE (82, 185)-(98, 185), 15  'STRAIGHT FACE

CIRCLE (544, 180), 20, 15        ' EDUCATOR
CIRCLE (537, 176), 3, 7       ' STUDENT LEFT EYE
CIRCLE (551, 176), 3, 7       ' STUDENT RIGHT EYE
LINE (536, 185)-(552, 185), 15  'STRAIGHT FACE

'---------------------------------
'draw, paint and get green circle
'---------------------------------
```

```
CIRCLE (clx%, cly%), 5, 10
PAINT (clx%, cly%), 10, 10
GET (clx% - 5, cly% - 5)-(clx% + 5, cly% + 5), getarray

hor1% = POINT(0)
ver1% = POINT(1)

PUT (415, 255), getarray

hor2% = POINT(0)
ver2% = POINT(1)

'---------------------------------
' draw boxes and big circles
'--------------------------------

lx1% = 105              'LIMIT
ly1% = 230              'LIMIT
lx2% = 260              'LIMIT
ly2% = 290              'LIMIT

rx1% = 380              'LIMIT
ry1% = 220              'LIMIT
rx2% = 570              'LIMIT
ry2% = 300              'LIMIT

trx1% = 480
try1% = 222

tlx1% = 190
tly1% = 220

COLOR 11
LINE (tlx1%, tly1%)-(tlx1%, tly1% - 48)      'top left_
 connect line
LINE (trx1%, try1%)-(trx1%, try1% - 50)      'top right_
 connect line
LINE (tlx1%, tly1% - 48)-(trx1%, try1% - 50) 'top
 horizontal_ connect line

COLOR 14
LINE (tlx1%, tly1% + 80)-(tlx1%, tly1% + 120) 'bot left
 connect line
LINE (trx1%, try1% + 80)-(trx1%, try1% + 118) 'bot right
 connect line
LINE (tlx1%, tly1% + 120)-(trx1%, try1% + 118)'bot_
 horizontal connect line

COLOR 15
LINE (rx1%, ry1%)-(rx2%, ry2%), 11, B   'right cyan box

LRCX% = 140
```

```
LRCY% = 260
RRCX% = 520

CIRCLE (LRCX%, LRCY%), 20, 12   'left red circle
CIRCLE (RRCX%, LRCY%), 20, 12   'right red circle

'draw beginning elastic lines

LINE (tlx1%, tryl%)-(hor1% - 5, ver1% - 5), 15       'draw_
 upper left line

LINE (tlx1%, tryl% + 75)-(hor1% - 5, ver1% - 5), 15  'draw_
 lower left line

LINE (trx1%, tryl%)-(hor2% + 5, ver2% + 5), 15       'draw_
 upper right line
LINE (trx1%, tryl% + 75)-(hor2% + 5, ver2% + 5), 15  'draw_
 lower right line

DO
LOOP UNTIL INKEY$ <> ""

'erase beginning elastic lines

LINE (tlx1%, tryl%)-(hor1% - 5, ver1% - 5), 0       'erase_
upper left line
LINE (tlx1%, tryl% + 75)-(hor1% - 5, ver1% - 5), 0  'erase_
lower left line

LINE (trx1%, tryl%)-(hor2% + 5, ver2% + 5), 0       'erase_
upper right line
LINE (trx1%, tryl% + 75)-(hor2% + 5, ver2% + 5), 0  'erase_
lower right line

'----------------------------------
'erase first and second dots
'----------------------------------

 PUT (hor1% - 10, ver1% - 10), getarray, XOR
 PUT (hor2%, ver2%), getarray, XOR

 LOCATE 19, 31                           'erase residue
 PRINT "  "
 LOCATE 19, 53                           'erase residue
 PRINT "  "

 speed% = 30000

'=============================
' Begin loop
'=============================
INC% = 1
```

```
FOR i% = 1 TO 150

GOSUB arrows

IF i% = 1 THEN
 PRESET (clx%, cly%)                      ' locate graphic
pixel
 ELSE
  PRESET (hor1%, ver1%)                   ' locate graphic
pixel

 PUT (hor1%, ver1%), getarray, XOR                 'erase dot
 LINE (tlx1%, tryl%)-(hor1%, ver1%), 0             'erase
upper_ left line
 LINE (tlx1%, tryl% + 75)-(hor1%, ver1%), 0        'erase
lower_ left line
 CIRCLE (RRCX%, LRCY%), 20, 12                'left red circle

 PUT (hor2%, ver2%), getarray, XOR                 'erase dot
 LINE (trx1%, tryl%)-(hor2%, ver2%), 0             'erase
upper_ left line
 LINE (trx1%, tryl% + 75)-(hor2%, ver2%), 0        'erase
lower_ left line
 CIRCLE (LRCX%, LRCY%), 20, 12               'right red circle

INC% = INC% + 1
IF INC% = 5 THEN
CIRCLE (LRCX%, LRCY%), 20, 0          'erase  left red circle
CIRCLE (RRCX%, LRCY%), 20, 0          'erase  right red
circle

LRCX% = LRCX% + 1
RRCX% = RRCX% - 1
CIRCLL (LRCX%, LRCY%), 20, 12

'left red circle
CIRCLE (RRCX%, LRCY%), 20, 12

'right red circle
INC% = 1
END IF

  PRESET (hor1%, ver1%)                   ' locate graphic
pixel

END IF

 RANDOMIZE TIMER
 pick1% = INT(18 * RND) - 9
 updn1% = SGN(pick1%) * 5
  IF ver1% <= ly1% + 25 THEN updn1% = 5
  IF ver1% >= ly2% - 30 THEN updn1% = -5
 RANDOMIZE TIMER
 pick1% = INT(18 * RND) - 10
 lfrt1% = SGN(pick1%) * 5
  IF hor1% <= lx1% + 10 THEN lfrt1% = 5
  IF hor1% >= lx2% - 0 THEN lfrt1% = -5

 PUT STEP(lfrt1%, updn1%), getarray       'print dot

 hor1% = POINT(0)
 ver1% = POINT(1)

''''''''''''''''''''''''''
'locate top of left line
''''''''''''''''''''''''''
LINE (tlx1%, try1%)-(hor1%, ver1%), 15 'draw upper left line
LINE (tlx1%, try1% + 75)-(hor1%, ver1%), 15 'draw lower
left_ line

'-----------------------
' smiley face section
'-----------------------

' both dots outside

IF hor1% < A% OR hor1% > B% OR ver1% < C% OR ver1% > D% THEN
IF hor2% < E% OR hor2% > F% OR ver2% < C% OR ver2% > D% THEN

LINE (78, 168)-(102, 189), 0, BF         'erase mouth
LINE (532, 168)-(556, 189), 0, BF         'erase mouth

CIRCLE (90, 180), 20, 15' STUDENT
CIRCLE (83, 176), 3, 7      ' STUDENT LEFT EYE
CIRCLE (97, 176), 3, 7      ' STUDENT RIGHT EYE
LINE (82, 185)-(98, 185), 15  'STRAIGHT FACE

CIRCLE (544, 180), 20, 15      ' EDUCATOR
CIRCLE (537, 176), 3, 7      ' STUDENT LEFT EYE
CIRCLE (551, 176), 3, 7      ' STUDENT RIGHT EYE
LINE (536, 185)-(552, 185), 15  'STRAIGHT FACE

'LINE (82, 185)-(98, 185), 15  'STRAIGHT FACE
'LINE (536, 185)-(552, 185), 15   'STRAIGHT FACE

END IF
END IF
'------------------
' both dots inside
'------------------
'LRCX% = 140
'LRCY% = 260
'RRCX% = 520
```

```
A% = LRCX% - 20
B% = LRCX% + 20
C% = LRCY% - 20
D% = LRCY% + 20
E% = RRCX% - 30
F% = RRCX% + 20

IF hor1% > A% AND hor1% < B% AND ver1% > C% AND ver1% < D%_
THEN
IF hor2% > E% AND hor2% < F% AND ver2% > C% AND ver2% < D%_
THEN
100
LINE (78, 168)-(102, 189), 0, BF          'erase mouth
LINE (532, 168)-(556, 189), 0, BF          'erase mouth

CIRCLE (90, 180), 20, 10             ' STUDENT
CIRCLE (83, 176), 4, 10, 0, PI!      ' STUDENT LEFT EYE
CIRCLE (97, 176), 4, 10, 0, PI!      ' STUDENT RIGHT EYE
CIRCLE (83, 176), 1, 10              ' STUDENT LEFT EYE
CIRCLE (97, 176), 1, 10              ' STUDENT RIGHT EYE

CIRCLE (544, 180), 20, 10             ' EDUCATOR
CIRCLE (537, 176), 3, 10, 0, PI!      ' STUDENT LEFT EYE
CIRCLE (551, 176), 3, 10, 0, PI!      ' STUDENT RIGHT EYE
CIRCLE (537, 176), 4, 10, 0, PI!      ' STUDENT LEFT EYE
CIRCLE (551, 176), 4, 10, 0, PI!      ' STUDENT RIGHT EYE

CIRCLE (90, 183), 8, 10, PI!, 0   'SMILE
CIRCLE (90, 183), 9, 10, PI!, 0   'SMILE
CIRCLE (544, 183), 8, 10, PI!, 0  'SMILE
CIRCLE (544, 183), 9, 10, PI!, 0  'SMILE

PUT (hor1%, ver1%), getarray, XOR         'erase dot
LINE (tlx1%, tryl%)-(hor1%, ver1%), 0  'erase upper left
line
LINE (tlx1%, tryl% + 75)-(hor1%, ver1%), 0 'erase lower
left_ line
CIRCLE (RRCX%, LRCY%), 20, 12             'left red circle
PUT (LRCX% - 5, LRCY% - 5), getarray, XOR 'draw left green_
dot
PUT (RRCX% - 5, LRCY% - 5), getarray, XOR 'draw right green
dot

GOTO 777
END IF
END IF

IF hor2% > E% AND hor2% < F% AND ver2% > C% AND ver2% < D%_
THEN

LINE (78, 168)-(102, 189), 0, BF          'erase mouth
LINE (532, 168)-(556, 189), 0, BF          'erase mouth
```

```
CIRCLE (90, 180), 20, 4       ' STUDENT
PUT (79, 174), aray7%         'squint

CIRCLE (544, 180), 20, 4       ' EDUCATOR
PUT (533, 174), aray7%         'squint

CIRCLE (90, 189), 8, 12, 0, PI! / 1 'FROWN
CIRCLE (544, 189), 8, 12, 0, PI! / 1 'FROWN
END IF

IF hor1% > A% AND hor1% < B% AND ver1% > C% AND ver1% < D%_
THEN

LINE (78, 168)-(102, 189), 0, BF          'erase mouth
LINE (532, 168)-(556, 189), 0, BF          'erase mouth

CIRCLE (90, 180), 20, 4       ' STUDENT

CIRCLE (83, 176), 3, 0       ' STUDENT LEFT EYE
CIRCLE (97, 176), 3, 0       ' STUDENT RIGHT EYE

PUT (79, 174), aray7%        'squint

CIRCLE (544, 180), 20, 4       ' EDUCATOR
PUT (533, 174), aray7%         'squint
CIRCLE (90, 189), 8, 12, 0, PI! / 1 'FROWN
CIRCLE (544, 189), 8, 12, 0, PI! / 1 'FROWN
END IF
'=================================================

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
IF i% = 1 THEN
 PRESET (415, 245)                             ' locate graphic
pixel
ELSE
  PRESET (hor2%, ver2%)                        ' locate graphic
pixel
END IF
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

 RANDOMIZE TIMER
 pick2% = INT(18 * RND) - 9
 updn2% = SGN(pick2%) * 5

'restrict dot

IF ver2% <= ry1% + 25 THEN updn2% = 5
IF ver2% >= ry2% - 25 THEN updn2% = -5

 RANDOMIZE TIMER
 pick2% = INT(18 * RND) - 7
 lfrt2% = SGN(pick2%) * 5
```

```
'restrict dot
  IF hor2% <= rx1% + 20 THEN lfrt2% = 5          'was +10
  IF hor2% >= rx2% - 20 THEN lfrt2% = -5         'was -10

 PUT STEP(lfrt2%, updn2%), getarray         'print dot

 hor2% = POINT(0)
 ver2% = POINT(1)

''''''''''''''''''''''''''
'locate top of right line
''''''''''''''''''''''''''

LINE (trx1%, try1%)-(hor2%, ver2%), 15  'draw upper right_
line
LINE (trx1%, try1% + 75)-(hor2%, ver2%), 15'draw lower
right_ line

  FOR 1% = 1 TO speed%
  NEXT 1%

GOSUB arrows

kk% = kk% + 1
IF kk% = 4 THEN kk% = 1

NEXT i%       '!!!!!!!  END OF BIG LOOP  !!!!!!!!!!!

PUT (hor2%, ver2%), getarray, XOR              'erase dot
LINE (trx1%, try1%)-(hor2%, ver2%), 0    'erase upper left_
line
LINE (trx1%, try1% + 75)-(hor2%, ver2%), 0 'erase lower
left_ line

GOTO 100

777

'successful completion

 COLOR 15

LINE (tlx1%, try1%)-(LRCX%, LRCY%), 15     'draw upper left_
line
LINE (tlx1%, try1% + 75)-(LRCX%, LRCY%), 15 'draw lower
left_ line

LINE (trx1%, try1%)-(RRCX%, LRCY%), 15    'draw upper right_
line
LINE (trx1%, try1% + 75)-(RRCX%, LRCY%), 15'draw lower
right_ line
```

```
555

LOCATE 11, 31
PRINT "Hit any key to CONTINUE"
DO
LOOP UNTIL INKEY$ <> ""

CLS
RUN "MAT20.EXE"
666
END
'=========================================
'---------
arrows:
'---------
lo% = 220
in% = 40
IF kk% = 1 THEN

PUT (473, tryl% - 8), arayl%        'top
PUT (183, tryl% - 8), aray2%         'top

PUT (183, tryl% + 80), aray5%        'bottom
PUT (473, tryl% + 80), aray6%         'bottom

PUT (lo% + 6 * in%, tlyl% - 52), aray3%
PUT (!o% + 3 * in%, tlyl% - 52), aray3%
PUT (lo% + 0 * in%, tlyl% - 52), aray3%
PUT (lo% + 6 * in%, tlyl% + 116), aray4%
PUT (lo% + 3 * in%, tlyl% + 116), aray4%
PUT (lo% + 0 * in%, tlyl% + 116), aray4%
COLOR 11
GOSUB colors

ELSEIF kk% = 2 THEN

PUT (183, tryl% + 100), aray5%
PUT (lo% + 6.3 * in%, tlyl% + 116), aray4%

PUT (473, tryl% - 28), arayl%
PUT (183, tryl% - 48), aray2%
PUT (lo% + 5 * in%, tlyl% - 52), aray3%
PUT (lo% + 2 * in%, tlyl% - 52), aray3%
PUT (lo% + 1 * in%, tlyl% + 116), aray4%
PUT (lo% + 4 * in%, tlyl% + 116), aray4%
COLOR 10
GOSUB colors

ELSEIF kk% = 3 THEN

PUT (473, tryl% + 110), aray6%
PUT (190, tlyl% + 116), aray4%
PUT (473, try1% - 48), aray1%
PUT (183, try1% - 28), aray2%
PUT (lo% + 1 * in%, tly1% - 52), aray3%
PUT (lo% + 4 * in%, tly1% - 52), aray3%
PUT (lo% + 2 * in%, tly1% + 116), aray4%
PUT (lo% + 5 * in%, tly1% + 116), aray4%
COLOR 14
GOSUB colors
END IF
RETURN
END
'-------
colors:
'--------
LOCATE 1, 18
PRINT "MEANINGFUL LEARNING"
LOCATE 3, 18
PRINT "    requires dialog,"
LOCATE 4, 18
PRINT "              exchange,"
LOCATE 5, 18
PRINT "               sharing, "
LOCATE 6, 18
PRINT "                and sometimes compromise."
LOCATE 8, 14
PRINT "
"
LOCATE 9, 14
PRINT "
"
RETURN
END

'====================================
'  MAIN PROGRAM
'====================================

DECLARE SUB BLINK2 (A%(), E!(), MC%, RC%, ROWA%, COLB%,_
ACBR%)
DECLARE SUB clr2 (R1%, C1%, R2%, C2%)
DECLARE SUB BLINK (A%(), B%(), MC%, RC%, ROWA%, COLB%,
ACBR%)
DECLARE SUB determ2 (i%, j%, A%(), name$, sol%)
DECLARE SUB iandj (i%, j%, hfc%, hbc%, ijr%, ijc%, ijfc%,_
ijbc%)
DECLARE SUB clr1 (R1%, C1%, R2%, C2%)
DECLARE SUB minor (i%, j%, A%(), name$, sol%, tsol%)
DECLARE SUB box (TLR%, TLC%, WIDE%, LL%, FGC%, BGC%, bx$,_
sfc%, style%, syn$)
DIM A%(1 TO 3, 1 TO 3), B%(1 TO 3, 1 TO 3), C%(1 TO 3, 1 TO_
3)
DIM D%(1 TO 3, 1 TO 3), E!(1 TO 3, 1 TO 3)
DIM item$(20)
COMMON SHARED item$(), TLCB%, TLCR%, TLRB%, TLRR%, TLRA%,_
TLCA%, BC1%, BR1%
COMMON A%, B%
COMMON SHARED ROWA%, COLB%, ACBR%

'CALL intro

CLEAR , , 1000                                  'Increase stack
size
SCREEN 0

''''''''''''''''''''''''''''''''''''''''''''''
' tlr% - Position of Top Left Row
' tlc% - Position of Top Left Column
' wide% - Width of longest line in list
' ll % - number of items in list
' fgc% - Foreground color
' bgc% - Background color
' bx$ - "n" display list/"y" don't display list
' sfc% - shadow foreground color
' style% - 1 for 2 lines/2 for matrix/3 for determinent
' syn$          "y" use a shadow - "n" no shadow
'
''''''''''''''''''''''''''''''''''''''''''''''
777
COLOR 15, 7
CLS
'-----------------------------------------------
 item$(1) = "M A T R I X   O P E R A T I O N S"
 CALL box(1, 20, 34, 1, 15, 1, "n", 0, 1, "y")
'-----------------------------------------------
 item$(1) = ""
 item$(2) = " 1.  ADDITION                     5.
TRANSPOSITION_ "
 item$(3) = " 2.  SUBTRACTION                  6.
MULTIPLICATION (DOT PRODUCT)"
 item$(4) = " 3.  SCALAR MULTIPLICATION        7.  INVERSION "
 item$(5) = " 4.  SCALAR DIVISION              8.  Quit"
 CALL box(6, 7, 63, 5, 15, 2, "n", 0, 1, "y")
 LOCATE 7, 38
PRINT "M E N U"
LOCATE 8, 8
PRINT
"╟──────────────────────────────────────────────────────╢"

'-----------------------------------------------
'-----------------------------------------------
 item$(1) = "ENTER THE NUMBER OF YOUR SELECTION"
 CALL box(21, 20, 34, 1, 15, 4, "n", 0, 1, "y")
'-----------------------------------------------
2
```

```
 DO
  SEL1$ = INKEY$
  LOOP UNTIL SEL1$ <> ""
   IF SEL1$ = "6" THEN GOSUB MULTIPLY
   IF SEL1$ = "1" THEN GOSUB ADD
   IF SEL1$ = "2" THEN GOSUB ADD
   IF SEL1$ = "5" THEN GOSUB TRANSPOSE
   IF SEL1$ = "7" THEN GOSUB INVERT
   IF SEL1$ = "3" THEN GOSUB scalar
   IF SEL1$ = "4" THEN GOSUB scalar
   IF SEL1$ = "Q" OR SEL1$ = "q" OR SEL1$ = "8" THEN GOTO
666
GOTO 2
'    ==========================================
'       G O S U B S   B E G I N   H E R E
'    ==========================================
'==================================================
scalar:         'scalar multiplication and division
'==================================================
202
cb% = 7                ' COLOR OF BACKGROUND
COLOR 15, cb%
CLS
IF SEL1$ = "3" THEN
'---------------------------------- ---
 item$(1) = "S C A L A R  M U L T I P L I C A T I O N"
 CALL box(1, 20, 41, 1, 15, 4, "n", 0, 1, "y")
'-------------------------------------
ELSE
'---------------------------------------------------
 item$(1) = "      S C A L A R  D I V I S I O N "
 CALL box(1, 20, 39, 1, 15, 4, "n", 0, 1, "y")
'---------------------------------------------------
END IF
404
'------------------------------------------------------------
item$(1) = "MATRICIES may have a maximum of 3 ROWS and 3
COLUMNS"
 item$(2) = " ENTER THE DIMENSIONS OF THE MATRIX"
 item$(3) = "                                    "
 item$(4) = "                                    "
 CALL box(7, 14, 52, 4, 15, 2, "n", 0, 1, "y")
'-----------------------------------------------------------
LOCATE 12, 24
 INPUT "How many ROWS in the Matrix: ", ROWA%
LOCATE 13, 24
 INPUT "How many COLUMNS in the Matrix: ", COLA%

IF ROWA% < 1 OR COLA% < 1 GOTO 404      ' limit matricies
IF ROWA% > 3 OR COLA% > 3 GOTO 404       ' to 3 x 3

LOCATE 7, 1                  ''''''''''''''''''''
```

```
COLOR 15, cb%                 '    clear
FOR i% = 1 TO 14              '
PRINT SPC(60);                '  ENTER BOX
NEXT i%                       '
                              ''''''''''''''''''''

'-------------------------
' LOCATION OF MATRIX BOXES
'------------------------------------------
 TLRA% = 10                     'Top Left Row A MATRIX
 TLCA% = 16                     'Top Left Column A MATRIX
 TLCB% = 30                     'Top Left Column B MATRIX
 TLRB% = 10                     'Top Left row B MATRIX
 TLCR% = 45                     'Top Left Column R MATRIX
 TLRR% = 10                     'Top Left Row R MATRIX
'------------------------------------------------------------
IF COLA% = 1 THEN                    'one column of A
  TLCA% = TLCA% + 13
  TLCB% = TLCB% + 7
  TLCR% = TLCR% + 2
  NLEN% = 1
ELSEIF COLA% = 2 THEN                'two columns of A
  TLCA% = TLCA% + 6
  TLCB% = TLCB% + 1
  TLCR% = TLCR% + 1
  NLEN% = 7
ELSEIF COLA% = 3 THEN                'three columns of A
  TLCB% = TLCB% + 1
  TLCR% = TLCR% + 2
  NLEN% = 13
END IF

CALL box(TLRA%, TLCA%, NLEN%, ROWA%, 15, 1, "y", 0, 2,"y")_
'matrix A
NLEN% = NLEN% - 2
IF SEL1$ = "3" THEN
CALL box(TLRR%, TLCR%, NLEN% + 1, ROWA%, 15, 2, "y", 0, 2,_
 "y")'matrix R
ELSE
CALL box(TLRR%, TLCR%, NLEN% + 2, ROWA%, 15, 2, "y", 0, 2,_
 "y")'matrix R
END IF

CALL box(22, 13, 50, 1, 15, 0, "y", cb%, 1, "y")'equation
box
COLOR 11, 0
LOCATE 23, 21
PRINT "Hit the <Space Bar> to watch the process"
COLOR 14, 0
LOCATE 23, 29
PRINT "<Space Bar>"

'----------------
```

```
' LOCATE OPERATORS
'----------------
 COLOR 15, cb%
 IF SEL1$ = "3" THEN
 LOCATE TLRA% + ROWA%, TLCA% + (NLEN% + 8): PRINT "x"
 ELSE
 LOCATE TLRA% + ROWA%, TLCA% + (NLEN% + 8): PRINT "÷"
 END IF
 LOCATE TLRA% + ROWA%, TLCA% + (NLEN% + 14): PRINT "="
'------------------------------------------------------------

FOR i% = 1 TO 3              ''''''''''
FOR j% = 1 TO 3              '
RANDOMIZE TIMER              '  pick random numbers for array
A
pick% = INT(18 * RND) - 9    '  between -9 and 9
A%(i%, j%) = pick%           '
NEXT j%                      '
NEXT i%                      ''''''''''

23

RANDOMIZE TIMER              '  pick random numbers for scalar
Dr% = INT(18 * RND) - 9      '  between -9 and 9 but not 0
IF Dr% = 0 GOTO 23           '
'------------------
' LABEL EACH MATRIX AND OPERATION (X/÷)
'------------------
 COLOR 7, 0                                     'COLOR OF LABEL

 LOCATE TLRA% + (ROWA% * 2) + 1, TLCA% + (.5 * NLEN% + 2)
 PRINT "A"
 IF SEL1$ = "3" THEN
 LOCATE TLRR% + (ROWA% * 2) + 1, TLCR% + (.5 * NLEN%) + 1
 PRINT "A  x "; Dr%
 ELSE
 LOCATE TLRR% + (ROWA% * 2) + 1, TLCR% + (.5 * NLEN%) + 1
 PRINT "A  ÷ "; Dr%
 END IF
'-------------------------------------------------------
 BC1% = 6                                  ' COLUMN SEPARATION
 BR1% = 2                                  ' ROW SEPARATION
'------------------------------------------------------
'++++++++++++++++++++++++++++++++++++++++++++++++

FOR i% = 1 TO ROWA%                        '
AR% = BR1% * i% + TLRA% - 1                '  FOR ARRAY
FOR j% = 1 TO COLA%                        '
AC% = BC1% * j% + TLCA% - 4                '
BR% = BR1% * j% + TLRB% - 1                '

COLOR 15, 1                                ' MATRIX A COLOR
```

```
LOCATE AR%, AC%                       '  PRINT
PRINT A%(i%, j%)                      '  ARRAY
'++++++++++++++++++++++
 NEXT j%                               '  LOOP
 NEXT i%                               '

'''''''''''''''''''''
' locate the scalar
'''''''''''''''''''''

sc% = 0                'scalar color

COLOR sc%, cb%
 LOCATE TLRA% + ROWA%, TLCA% + (NLEN% + 10): PRINT Dr%
COLOR 15, 1

DO
LOOP UNTIL INKEY$ <> ""

COLOR 30, cb%
 LOCATE TLRA% + ROWA%, TLCA% + (NLEN% + 10): PRINT Dr%
COLOR 15, 1

BLNKC% = 14                           ' BLINK COLOR

FOR i% = 1 TO ROWA%                   '
AR% = BR1% * i% + TLRA% - 1
FOR j% = 1 TO COLA%                   '
AC% = BC1% * j% + TLCA% - 4           '

BR% = AR%         '
BC% = BC1% * j% + TLCB% - 4          '

'------------------------------------------------------
COLOR 30

LOCATE AR%, AC%                       '  BLINK
PRINT A%(i%, j%)                      '  A ARRAY ELEMENTS

COLOR 15, 1
RC% = BC1% * j% + TLCR% - 4           '
  LOCATE BR%, RC%                     ' PRINT
  PRINT "█"                           ' RESULT BOX
IF SEL1$ = "3" THEN
S% = A%(i%, j%) * Dr%
ELSE
S! = A%(i%, j%) / Dr%
END IF
'-------------------------------------------------------
COLOR 11, 0        ' clear equation box at bottom
LOCATE 23, 40      ' of screen
```

```
PRINT SPC(24);
'-------------------------------------------------------------

   COLOR 15, 0
   LOCATE 23, 45
'''''''''''
IF SEL1$ = "3" THEN
   PRINT USING "(##  x ##) = █"; A%(i%, j%); D%;   ' PRINT
ELSE
   PRINT USING "(##  ÷ ##) = █"; A%(i%, j%); D%;   ' PRINT
END IF
DO
LOOP WHILE INKEY$ = ""

COLOR 15, 1                                    '  TURN OFF BLINK
LOCATE AR%, AC%                                '  PRINT
PRINT A%(i%, j%)                               '  NON BLINKING A ARRAY
ELEMENT

  COLOR 15, 2
  LOCATE BR%, RC%                              ' PRINT
IF SEL1$ = "3" THEN
  PRINT S%                                     ' result
ELSE
  PRINT USING "##.##"; S!
 END IF

COLOR 15, 1

NEXT j%
NEXT i%

'''''''''''''''''''''''''
'turn off blinking scalar
'
'''''''''''''''''''''''''
COLOR sc%, cb%
 LOCATE TLRA% + ROWA%, TLCA% + (NLEN% + 10): PRINT Dr%

'=================
GOSUB escape
'==================

DO
LOOP UNTIL INKEY$ <> ""
GOTO 666

'=============================================================
ADD:                                          'matrix addition and
subtraction
'=============================================================
200
```

```
cb% = 7                ' COLOR OF BACKGROUND
COLOR 15, cb%
CLS
IF SEL1$ = "2" THEN
'---------------------------------- ---
 item$(1) = "  M A T R I X   S U B T R A C T I O N"
 CALL box(1, 20, 39, 1, 15, 4, "n", 0, 1, "y")
'--------------------------------------
ELSEIF SEL1$ = "1" THEN
'---------------------------------- ---
 item$(1) = "   M A T R I X   A D D I T I O N "
 CALL box(1, 20, 39, 1, 15, 4, "n", 0, 1, "y")
'--------------------------------------
END IF
444
'--------------------------------------
item$(1) = "MATRICIES may have a maximum of 3 ROWS and 3
COLUMNS"
 item$(2) = " ENTER THE DIMENSIONS OF THE MATRIX"
 item$(3) = "                                     "
 item$(4) = "                                     "
 CALL box(7, 14, 52, 4, 15, 2, "n", 0, 1, "y")
'--------------------------------------


LOCATE 12, 24
 INPUT "How many ROWS in the Matrix: ", ROWA%
LOCATE 13, 24
 INPUT "How many COLUMNS in the Matrix: ", COLA%

IF ROWA% < 1 OR COLA% < 1 GOTO 444          ' limit matricies
IF ROWA% > 3 OR COLA% > 3 GOTO 444                ' to 3 x 3

LOCATE 7, 1               '''''''''''''''''''
COLOR 15, cb%             '    clear
FOR i% = 1 TO 14          '
PRINT SPC(60);            '  ENTER BOX
NEXT i%                   '
                          '''''''''''''''''''
'-------------------------
' LOCATION OF MATRIX BOXES
'------------------------------------------------------------
 TLRA% = 10                'Top Left Row A MATRIX
 TLCA% = 5                 'Top Left Column A MATRIX
 TLCB% = 30                'Top Left Column B MATRIX
 TLRB% = 10                'Top Left row B MATRIX
 TLCR% = 55                'Top Left Column R MATRIX
 TLRR% = 10                'Top Left Row R MATRIX
'------------------------------------------------------------
IF COLA% = 1 THEN               'one column of A
  TLCA% = TLCA% + 20
  TLCB% = TLCB% + 10
  NLEN% = 1
```

```
ELSEIF COLA% = 2 THEN           'two columns of A
  TLCA% = TLCA% + 10
  TLCB% = TLCB% + 5
  NLEN% = 7
ELSEIF COLA% = 3 THEN           'three columns of A
  TLCB% = TLCB% + 1
  TLCR% = TLCR% + 2
  NLEN% = 13
END IF

CALL box(TLRA%, TLCA%, NLEN%, ROWA%, 15, 1, "y", 0, 2, "y")_
'matrix A
NLEN% = NLEN% - 2
CALL box(TLRB%, TLCB%, NLEN%, ROWA%, 15, 1, "y", 0, 2,_
"y")'matrix B
NLEN% = NLEN% - 1
CALL box(TLRR%, TLCR%, NLEN%, ROWA%, 15, 2, "y", 0, 2,
"y")'matrix R

'------------------------------------------------------------
CALL box(22, 13, 50, 1, 15, 0, "y", cb%, 1, "y")'equation
box
COLOR 11, 0
LOCATE 23, 21
PRINT "Hit the <Space Bar> to watch the process"
COLOR 14, 0
LOCATE 23, 29
PRINT "<Space Bar>"
'------------------------------------------------------------

'------------------
' LABEL EACH MATRIX AND OPERATION (+/-)
'------------------
 COLOR 7, 0                                 'COLOR OF LABEL

 LOCATE TLRA% + (ROWA% * 2) + 1, TLCA% + (.5 * NLEN%)
 PRINT "A"
 LOCATE TLRB% + (ROWA% * 2) + 1, TLCB% + (.5 * NLEN%)
 PRINT "B"
 IF SEL1$ = "3" THEN
 LOCATE TLRR% + (ROWA% * 2) + 1, TLCR% + (.5 * NLEN%) - 1:
PRINT "A - B"
 ELSE
 LOCATE TLRR% + (ROWA% * 2) + 1, TLCR% + (.5 * NLEN%) - 1:
PRINT "A + B"
 END IF
'----------------
' LOCATE OPERATORS
'----------------
 COLOR 15, cb%
 IF SEL1$ = "2" THEN
 LOCATE TLRA% + ROWA%, TLCA% + (NLEN% + 6): PRINT "-"
 ELSEIF SEL1$ = "1" THEN
 LOCATE TLRA% + ROWA%, TLCA% + (NLEN% + 6): PRINT "+"
 END IF
 LOCATE TLRB% + ROWA%, TLCB% + (NLEN% + 6): PRINT "="
'-----------------------------------------------------------

FOR i% = 1 TO 3                  ''''''''''
FOR j% = 1 TO 3                  '
RANDOMIZE TIMER                  '  pick random numbers for array
A
pick% = INT(18 * RND) - 9        '  between -9 and 9
A%(i%, j%) = pick%               '
NEXT j%                          '
NEXT i%                          ''''''''
FOR i% = 1 TO 3                  ''''''''''
FOR j% = 1 TO 3                  '
RANDOMIZE TIMER                  '  pick random numbers for array
B
pick% = INT(18 * RND) - 9        '  between -9 and 9
B%(i%, j%) = pick%               '
NEXT j%                          '
NEXT i%                          ''''''''
'-----------------------------------------------------------
 BC1% = 6                                ' COLUMN SEPARATION
 BR1% = 2                                ' ROW SEPARATION
'-------------------------------------------------------
'+++++++++++++++++++++++++++++++++++++++++++++++++

FOR i% = 1 TO ROWA%                      '
AR% = BR1% * i% + TLRA% - 1              '  FOR ARRAY
FOR j% = 1 TO COLA%                      '
AC% = BC1% * j% + TLCA% - 4              '
BR% = BR1% * j% + TLRB% - 1              '
                                         '
COLOR 15, 1                              ' MATRIX A COLOR
LOCATE AR%, AC%                          '  PRINT
PRINT A%(i%, j%)                         '  ARRAY
'+++++++++++++++++++++
BR% = AR%
BC% = (TLCB% + 2) + ((j% - 1) * 6)
LOCATE BR%, BC%                     'LOCATE B MATRIX ELEMENTS
PRINT B%(i%, j%)                    '
NEXT j%                             '  LOOP
NEXT i%                             '
DO
LOOP UNTIL INKEY$ <> ""

BLNKC% = 14                             ' BLINK COLOR

FOR i% = 1 TO ROWA%
AR% = BR1% * i% + TLRA% - 1
FOR j% = 1 TO COLA%
```

```
AC% = BC1% * j% + TLCA% - 4

BR% = AR%
BC% = BC1% * j% + TLCB% - 4
'-------------------------------------------------------

COLOR 30

LOCATE AR%, AC%                                 '  BLINK
PRINT A%(i%, j%)                                '  A ARRAY ELEMENTS

LOCATE BR%, BC%                                 '  BLINK
PRINT B%(i%, j%)                                '  B ARRAY ELEMENTS

COLOR 15, 1
RC% = BC1% * j% + TLCR% - 4                     '
  LOCATE BR%, RC%                               ' PRINT
  PRINT "█"                                     ' RESULT BOX

IF SEL1$ = "2" THEN                             '
 S% = A%(i%, j%) - B%(i%, j%)                   ' PRINT ANSWER
  ELSEIF SEL1$ = "1" THEN                       ' PRINT ANSWER
 S% = A%(i%, j%) + B%(i%, j%)                   '
END IF

'--------------------------------------------------------
COLOR 11, 0            'clear equation box at bottom
LOCATE 23, 40          ' of screen
PRINT SPC(24);
'--------------------------------------------------------

   COLOR 15, 0
   LOCATE 23, 45
'''''''''''
IF SEL1$ = "3" THEN
   PRINT USING "(##  - ##) =  █"; A%(i%, j%); B%(i%, j%);
' PRINT
ELSE
   PRINT USING "(##  + ##) =  █"; A%(i%, j%); B%(i%, j%);
' PRINT
END IF
DO
LOOP WHILE INKEY$ = ""


COLOR 15, 1                                     '  TURN OFF BLINK
LOCATF AR%, AC%                                 '  PRINT
PRINT A%(i%, j%)                                '  NON BLINKING A ARRAY
ELEMENT

LOCATE BR%, BC%                                 '  BLINK
PRINT B%(i%, j%)                                '  B ARRAY ELEMENTS
```

```
  COLOR 15, 2
  LOCATE BR%, RC%                                   ' PRINT
  PRINT S%                                          ' SUM or DIFF OF
ELEMENTS
  COLOR 15, 1

NEXT j%
NEXT i%

GOSUB escape

'===========================================================
TRANSPOSE:
'===========================================================

33
bclr% = 7

COLOR 15, bclr%
CLS
'---------------------------------------
 item$(1) = "M A T R I X   T R A N S P O S I T I O N"
 CALL box(1, 20, 39, 1, 15, 2, "n", 0, 1, "y")
'---------------------------------------
44
'---------------------------------------
item$(1) = "MATRICIES may have a maximum of 3 ROWS and 3
COLUMNS"
 item$(2) = " ENTER THE DIMENSIONS OF THE MATRIX"
 item$(3) = "                                      "
 item$(4) = "                                      "
 item$(5) = "                                      "
 CALL box(7, 14, 52, 5, 15, 1, "n", 0, 1, "y")
'---------------------------------------
LOCATE 12, 24
 INPUT "How many ROWS in the Matrix: ", ROWA%
LOCATE 13, 24
 INPUT "How many COLUMNS in the Matrix: ", COLA%
IF ROWA% < 1 OR COLA% < 1 GOTO 44     ' limit matricies
IF ROWA% > 3 OR COLA% > 3 GOTO 44     ' to 3 x 3

89

 COLOR 15, 0
LOCATE 18, 28
 PRINT "Transpose <R>ows or <C>olumns?"
LOCATE 18, 38
COLOR 14
 PRINT ; "<R>"; : COLOR 11: PRINT ; "ows";
COLOR 14
LOCATE 18, 48
 PRINT ; "<C>"; : COLOR 11: PRINT ; "olumns";
 DO
 RC$ = INKEY$
 LOOP UNTIL RC$ <> ""

IF RC$ = "r" OR RC$ = "R" THEN
LOCATE 21, 35
COLOR 0, 7: PRINT "ROWS to COLUMNS": COLOR 15, 1
GOTO 91
ELSEIF RC$ = "c" OR RC$ = "C" THEN
LOCATE 21, 35
COLOR 0, 7: PRINT "COLUMNS to ROWS": COLOR 15, 1
GOTO 91
END IF
GOTO 89

91

LOCATE 7, 1                 '''''''''''''''''''
COLOR 15, bclr%             '  clear
FOR i% = 1 TO 16            '
PRINT SPC(60);              '  ENTER BOX
NEXT i%                     '
                            '''''''''''''''''''

'-------------------------
' LOCATION OF MATRIX BOXES
'-----------------------------------------------------------
 TLRA% = 10                   'Top Left Row A MATRIX
 TLCA% = 20                   'Top Left Column A MATRIX
 TLCB% = 43                   'Top Left Column B MATRIX
 TLRB% = 10                   'Top Left Column B MATRIX
'-----------------------------------------------------------
IF COLA% = 1 THEN                  'one column of A
  MLEN% = 1
  TLCA% = TLCA% + 10
ELSEIF COLA% = 2 THEN              'two columns of A
 MLEN% = 7
  TLCA% = TLCA% + 6
ELSEIF COLA% = 3 THEN              'three columns of A
 MLEN% = 13
END IF
IF ROWA% = 1 THEN                  'one ROW of A
  NLEN% = 1
ELSEIF ROWA% = 2 THEN              'two ROWS of A
 NLEN% = 7
ELSEIF ROWA% = 3 THEN              'three ROWS of A
 NLEN% = 13
END IF
CALL box(TLRA%, TLCA%, MLEN%, ROWA%, 15, 1, "y", 0, 2,_
"y")'matrix A
CALL box(TLRB%, TLCB%, NLEN%, COLA%, 15, 4, "y", 0, 2,_
"y")'matrix AT
```

```
'-----------------
' LABEL EACH MATRIX
'-----------------

 COLOR 7, 0                              'COLOR OF LABEL
 LLRB% = TLRB% + (COLA% * 2) + 1   'Location of Lable B
(row)
 LLCB% = TLCB% + (.5 * NLEN%)   'Location of Lable B
(column)
 LLRA% = TLRA% + (ROWA% * 2) + 1   'Location of Lable A
(row)
 LLCA% = TLCA% + (.5 * MLEN% + 1)'Location of Lable
A(column)

 LOCATE LLRB%, LLCB%: PRINT "(A) T"    'lable B
 LOCATE LLRA%, LLCA%: PRINT "(A)"      'lable A

'--------------
 COLOR 15, MC%
'------------------------------------------------------------
COLOR 15, 1
FOR i% = 1 TO 3              '''''''''
FOR j% = 1 TO 3              '
RANDOMIZE TIMER              '  pick random numbers for array
A
pick% = INT(10 * RND) - 5    '  between -5 and 5
A%(i%, j%) = pick%           '
NEXT j%                      '
NEXT i%                      ''''''''
'------------------------------------------------------------
 BC1% = 6                                ' COLUMN SEPARATION
 BR1% = 2                                ' ROW SEPARATION
'-----------------------------------------------------
FOR i% = 1 TO ROWA%                      '
AR% = BR1% * i% + TLRA% - 1              '  FOR ARRAY
FOR j% = 1 TO COLA%                      '
AC% = BC1% * j% + TLCA% - 4              '
BR% = BR1% * j% + TLRB% - 1              '

LOCATE AR%, AC%                          '  PRINT
PRINT A%(i%, j%)                         '  ARRAY
NEXT j%                                  '
NEXT i%                                  '


'------------------------------------------------------------
cb% = 7                                              'BACKGROUND
COLOR
CALL box(22, 13, 50, 1, 15, 0, "y", cb%, 1, "y")
'equation box
COLOR 11, 0
LOCATE 23, 21
```

```
PRINT "Hit the <Space Bar> to watch the process"
COLOR 14, 0
LOCATE 23, 29
PRINT "<Space Bar>"
'---------------------------------------------------------
COLOR 15, 1


DO
LOOP WHILE INKEY$ = ""

' Perform the transposition

IF RC$ = "r" OR RC$ = "R" THEN
BLNKC% = 14                           ' BLINK COLOR
FOR i% = 1 TO ROWA%                   '
AR% = BR1% * i% + TLRA% - 1           '  FOR ARRAY
BC% = TLCB% + 2 + ((i% - 1) * 6)

FOR j% = 1 TO COLA%                   '
AC% = BC1% * j% + TLCA% - 4           '
BR% = BR1% * j% + TLRB% - 1           '

LOCATE AR%, AC%                       '  BLINK
COLOR 30                              '  THE
PRINT A%(i%, j%)                      '  ARRAY ELEMENT

  COLOR 15
  LOCATE BR%, BC%                      ' PRINT
  PRINT "█"                           ' RESULT BOX


DO
LOOP WHILE INKEY$ = ""
COLOR 15, 1                           '  TURN OFF BLINK
LOCATE AR%, AC%                       '  PRINT
PRINT A%(i%, j%)                      '  NON BLINKING ARRAY
ELEMENT
  COLOR 15, 4
  LOCATE BR%, BC%                ' PRINT
  PRINT A%(i%, j%)               ' TRANSPOSED ELEMENT
  COLOR 15, 1
NEXT j%
NEXT i%

ELSEIF RC$ = "c" OR RC$ = "C" THEN

BLNKC% = 14                           ' BLINK COLOR
FOR j% = 1 TO COLA%                   '
 AC% = BC1% * j% + TLCA% - 4          '
 BR% = BR1% * j% + TLRB% - 1          '
```

```
FOR i% = 1 TO ROWA%                          '
 AR% = BR1% * i% + TLRA% - 1                 '  FOR ARRAY
 BC% = TLCB% + 2 + ((i% - 1) * 6)

LOCATE AR%, AC%                              '  BLINK
COLOR 30                                     '  THE
PRINT A%(i%, j%)                             '  ARRAY ELEMENT

  COLOR 15
  LOCATE BR%, BC%                             ' PRINT
  PRINT "█"                                  ' RESULT BOX


DO
LOOP WHILE INKEY$ = ""

COLOR 15, 1                                  '  TURN OFF BLINK
LOCATE AR%, AC%                              '  PRINT
PRINT A%(i%, j%)                             '  NON BLINKING ARRAY
ELEMENT
  COLOR 15, 4
  LOCATE BR%, BC%                            ' PRINT
  PRINT A%(i%, j%)                           ' TRANSPOSED ELEMENT
  COLOR 15, 1
NEXT i%
NEXT j%

END IF
GOSUB escape

 RETURN
 END

'===========================================================
INVERT:            'MATRIX INVERSION MENU
'===========================================================

COLOR 15, 7
CLS
'-----------------------------------------------------
 item$(1) = "    M A T R I X   I N V E R S I O N"
 CALL box(1, 20, 39, 1, 15, 2, "n", 0, 1, "y")
'-----------------------------------------------------
TR% = 6
TC% = 7
 CALL box(TR%, TC%, 66, 6, 15, 1, "y", 0, 1, "y")
LOCATE TR% + 1, TC% + 4
PRINT "There are several steps required to INVERT a MATRIX."
LOCATE TR% + 3, TC% + 4
PRINT "The next screen presents a menu from which the user_
may select"
LOCATE TR% + 4, TC% + 4
```

```
PRINT "a particular step.  Since the steps are listed in
the_ appropriate"
LOCATE TR% + 5, TC% + 4
PRINT "sequence for the user to learn the process of MATRIX_
INVERSION,"
LOCATE TR% + 6, TC% + 4
PRINT "we suggest that the user procede from one step to
the_ next."
LOCATE TR% + 7, TC% + 4
PRINT "Review a step at any time."
LOCATE TR% + 9, TC% + 14
PRINT "Hit any key to continue"

DO
LOOP UNTIL INKEY$ <> ""

98
deta% = 0  'initialize the determinant of A

TR% = 8
TC% = 16

COLOR 4, 2
CALL box(TR%, TC%, 45, 7, 15, 4, "y", 0, 1, "y")   'box for
matrix menu

LOCATE TR% + 1, TC% + 14
PRINT "MATRIX INVERSION MENU"

LOCATE TR% + 2, TC% + 1
PRINT "╟───────────────────────────────────────────╢"

LOCATE TR% + 4, TC% + 4
PRINT "1.  DETERMINANT of a MATRIX (2 x 2 Example)"

LOCATE TR% + 5, TC% + 4
PRINT "2.  DETERMINANT of a MATRIX (3 x 3 Example)"

LOCATE TR% + 7, TC% + 4
PRINT "3.  The COFACTOR MATRIX  "
LOCATE TR% + 8, TC% + 4
PRINT "    The ADJOINT MATRIX"
LOCATE TR% + 9, TC% + 4
PRINT "    The INVERTED MATRIX "

LOCATE TR% + 13, TC% + 6
PRINT "     (M)ain menu      (Q)uit"
COLOR 14
LOCATE TR% + 13, TC% + 11
PRINT "(M)"
LOCATE TR% + 13, TC% + 28
PRINT "(Q)"
```

```
COLOR 15, 0
LOCATE TR% + 15, TC% + 14
PRINT " Make your selection "

99 DO
sel$ = INKEY$
LOOP UNTIL sel$ <> ""
IF sel$ = "M" OR sel$ = "m" GOTO 777
IF sel$ = "Q" OR sel$ = "q" GOTO 666
IF sel$ = "1" THEN
 o% = 2
 GOSUB determin
 END IF
IF sel$ = "2" THEN
 o% = 3
 GOSUB determin
 END IF
IF sel$ = "3" THEN GOSUB COFACTOR
GOTO 99

 RETURN
 END
'===========================================================
MULTIPLY:
'===========================================================
1                              '   return here to continue
'* * PROGRAMMER * * * * * * * * * * * * * * * * * * * * * *
COLOR 15, 7       ' STARTING FOREGROUND/BACKGROUND COLOR   *
CLS               '                                        *
MC% = 4           'A and B matrix color (background)       *
RC% = 2           'RESULT  matrix color (background)       *
'* * * * * * * * * * * * * * * * * * * * * * * * * * * * *

'-------------------------------------
 item$(1) = "MATRIX MULTIPLICATION - THE DOT PRODUCT"
 CALL box(1, 20, 39, 1, 15, 1, "n", 0, 1, "y")
'-------------------------------------
'++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
'        This section is for checking the final answer
'++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

IF sel2$ = "v" THEN
'-------------------------------------                 '
 item$(1) = "VERIFICATION BY USING - THE DOT PRODUCT"
 CALL box(1, 20, 39, 1, 15, 1, "n", 0, 1, "y")
'-------------------------------------

MC% = 4             'A and B matrix color (background)
*
RC% = 2             'RESULT  matrix color (background)
*
```

```
ROWA% = 3
COLA% = 3
ROWB% = 3
COLB% = 3
'------------------------
' LOCATION OF MATRIX BOXES
'------------------------------------------------------------------
 TLCB% = 43                   'Top Left Column B MATRIX    keep
 TLCR% = 43                   'Top Left Column R MATRIX    keep
 TLCA% = -6 * COLA% + 37      'Top Left Column A MATRIX
 TLRR% = 15                   'Top Left Row R MATRIX       keep
 TLRA% = 15                   'Top Left Row A MATRIX       keep
'------------------------------------------------------------------
 MLEN% = 15
 TLRB% = 6
 NLEN% = 13

CALL box(TLRB%, TLCB%, NLEN%, ROWB%, 15, MC%, "y", 0,
2,"y")_ 'matrix A
CALL box(TLRA%, TLCA%, MLEN%, ROWA%, 15, MC%, "y", 0, 2,
"y") 'matrix A inv
CALL box(TLRR%, TLCR%, NLEN% - 2, ROWA%, 15, RC%, "y", 0,
2,_ "y")'Result matix

'-----------------------------------------------------------------------------------

 BC1% = 6                                ' COLUMN SEPARATION
 BR1% = 2                                ' ROW SEPARATION
'-------------------------------------------------------------
FOR i% = 1 TO 3                           '
                                          '
 AR% = BR1% * i% + TLRA% - 1              '  FOR ARRAYS
 BR% = BR1% * i% + TLRB% - 1              '''''''
                                          '
FOR j% = 1 TO 3                           '
                                          '
 AC% = BC1% * j% + TLCA% - 4              '
 BC% = BC1% * j% + TLCB% - 4              '  CALCULATE LOCATIONS
'-------------------------------------------------------------
COLOR 15, 4
LOCATE AR%, AC%                          '  PRINT
 PRINT USING "##.##"; E!(i%, j%)
LOCATE BR%, BC%                          '
PRINT A%(i%, j%)                         '  ARRAYS

NEXT j%                                  '  LOOP
NEXT i%                                  '

DO
LOOP WHILE INKEY$ = ""

'--------------------------------
CALL BLINK2(A%(), E!(), MC%, RC%, 3, 3, 3)
```

```
'--------------------------------

COLOR 1, 7                                       ''''''''''''''''''''
LOCATE 22, 1                                     ' Erase
PRINT "                      "                   ' Words
PRINT "                      "                   '
PRINT "                      ";                  ''''''''''''''''''''

'---------------------------
'  CONTINUE OR QUIT SEQUENCE
'---------------------------
CALL box(8, 3, 27, 1, 15, 0, "y", 7, 1, "y")
COLOR 11, 0
LOCATE 9, 7
PRINT "(A)gain - (M)enu - (Q)uit";
COLOR 14, 0
LOCATE 9, 7: PRINT "(A)";
LOCATE 9, 17: PRINT "(M)";
LOCATE 9, 26: PRINT "(Q)";
911 DO
sel$ = INKEY$
LOOP WHILE sel$ = ""
IF sel$ = "A" OR sel$ = "a" THEN GOTO 1
IF sel$ = "M" OR sel$ = "m" THEN GOTO 777
IF sel$ = "Q" OR sel$ = "q" THEN GOTO 666
GOTO 911

END IF

' end of check the answer
'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
4
'---------------------------------------
 item$(2) = " ENTER THE DIMENSIONS OF THE MATRICIES "
 item$(1) = "MATRICIES may have a maximum of 3 ROWS and 3
COLUMNS"
 item$(3) = "                                        "
 item$(4) = "                                        "
 item$(5) = "                                        "
 CALL box(7, 14, 52, 5, 15, 4, "n", 0, 1, "y")
'---------------------------------------

LOCATE 12, 24
 INPUT "How many ROWS in Matrix A: ", ROWA%
LOCATE 13, 24
 INPUT "How many COLUMNS in Matrix A: ", COLA%
LOCATE 14, 24
 INPUT "How many ROWS in Matrix B: ", ROWB%
LOCATE 15, 24
 INPUT "How many COLUMNS in Matrix B: ", COLB%
```

```
IF ROWA% < 1 OR COLA% < 1 OR COLB% < 1 OR ROWB% < 1 GOTO 4_
'limit matricies
IF ROWA% > 3 OR COLA% > 3 OR COLB% > 3 OR ROWB% > 3 GOTO 4_
'to 3 x 3
'-------------------------
' LOCATION OF MATRIX BOXES
'----------------------------------------------------------
 TLCB% = 43                 'Top Left Column B MATRIX
 TLCR% = 43                 'Top Left Column R MATRIX
 TLCA% = -6 * COLA% + 37    'Top Left Column A MATRIX
 TLRR% = 15                 'Top Left Row R MATRIX
 TLRA% = 15                 'Top Left Row A MATRIX
'----------------------------------------------------------
LOCATE 7, 1                ''''''''''''''''''''
COLOR 15, 7                '  clear
FOR i% = 1 TO 18           '
PRINT SPC(60);             '  ENTER BOX
NEXT i%                    '

'===========================================================
IF COLA% = 1 THEN       'one column of A and one row of B
  MLEN% = 1
ELSEIF COLA% = 2 THEN 'two columns of A and two rows of B
 MLEN% = 7
ELSEIF COLA% = 3 THEN 'three columns of A and three rows of
B
 MLEN% = 13
END IF
'--------------------------------
IF ROWB% = 1 THEN
  TLRB% = 10
ELSEIF ROWB% = 2 THEN
 TLRB% = 8
ELSEIF ROWB% = 3 THEN
 TLRB% = 6
END IF
'------------------------------
IF COLB% = 1 THEN
  NLEN% = 1
ELSEIF COLB% = 2 THEN
 NLEN% = 7
ELSEIF COLB% = 3 THEN
 NLEN% = 13
END IF

CALL box(TLRB%, TLCB%, NLEN%, ROWB%, 15, MC%, "y", 0,
2,"y")_   'matrix B
CALL box(TLRA%, TLCA%, MLEN%, ROWA%, 15, MC%, "y", 0, 2,
"y") 'matrix A
CALL box(TLRR%, TLCR%, NLEN% - 1, ROWA%, 15, RC%, "y", 0,
2,_ "y")'Result matix
```

```
'-----------------
' LABEL EACH MATRIX
'-----------------
 COLOR 7, 0                                  'COLOR OF LABEL
 LOCATE 13, TLCB% + (.5 * NLEN% + 1): PRINT "B"
 LOCATE TLRA% + (ROWA% * 2) + 1, TLCA% + (.5 * MLEN% + 1)
 PRINT "A"'lable A
LOCATE TLRR% + (ROWA% * 2) + 1, TLCB% + (.5 * NLEN% + 1) - 2
PRINT "A • B"
'---------------------

IF COLA% <> ROWB% THEN
TR% = 6
TC% = 4
CALL box(TR%, TC%, 26, 3, 11, 0, "y", 7, 1, "y")
COLOR 14, 0
LOCATE TR% + 3, TC% + 3
PRINT "  Matrix A and Matrix B"
LOCATE TR% + 4, TC% + 3
PRINT "  cannot be multiplied."
COLOR 11, 0
LOCATE TR% + 6, TC% + 3
PRINT "┤                        ├"
LOCATE TR% + 6, TC% + 4
COLOR 15
PRINT "Hit any key to try again"
LOCATE TR% + 1, TC% + 4
COLOR 28, 0
PRINT " STOP!   STOP!   STOP! "

COLOR 15, 4
'-------------------------------------------------------------
 BC1% = 6                               ' COLUMN SEPARATION
 BR1% = 2                               ' ROW SEPARATION
'-------------------------------------------------------
FOR 1% = 1 TO COLB%                     '
BC% = BC1% * 1% + TLCB% - 4             '  CALCULATE LOCATIONS
FOR i% = 1 TO ROWA%                     '
AR% = BR1% * i% + TLRA% - 1             '  FOR ARRAYS
FOR j% = 1 TO COLA%                     '
AC% = BC1% * j% + TLCA% - 4             '
FOR K% = 1 TO ROWB%                     '
BR% = BR1% * K% + TLRB% - 1             '''''''
'-------------------------------------------------------
LOCATE AR%, AC%                         '  PRINT
PRINT " X"                              '  x x x
LOCATE BR%, BC%                         '
PRINT " X"                              '
NEXT K%                                 '  LOOP
NEXT j%                                 '  LOOP
NEXT i%                                 '
NEXT 1%                                 '''''''''
```

```
COLOR 15, 0

 DO
 LOOP UNTIL INKEY$ <> ""

 GOTO 1
 END IF
                                                ''''''''''''''''''''
COLOR 0, 7                                      ' Watch the
LOCATE 22, 2                                    '
PRINT "Watch the"                               ' equation
LOCATE 23, 2                                    '
PRINT " EQUATION"                               ' here
LOCATE 24, 2                                    '
PRINT "     here  >>";                          '''''''''''''''''''

'-------------------------------------
IF COLA% = ROWB% THEN ACBR% = COLA%
'-------------------------------------

'---------------
 COLOR 15, MC%
'-----------------------------------------------------------------
FOR i% = 1 TO 3                                ''''''''''
FOR j% = 1 TO 3                                ' PICK RANDOM NUMBERS
RANDOMIZE TIMER                                '   for array A
pick% = INT(10 * RND) - 5                      '  between -5 and 5
A%(i%, j%) = pick%                             '
NEXT j%                                        '
NEXT i%                                        ''''''''
FOR i% = 1 TO 3                                ''''''''''
FOR j% = 1 TO 3                                'PICK RANDOM NUMBERS
RANDOMIZE TIMER                                '   for array B
pick% = INT(10 * RND) - 5                      '  between -5 and 5
B%(i%, j%) = pick%                             '
NEXT j%                                        '
NEXT i%                                        ''''''''
'-----------------------------------------------------------------
 BC1% = 6                                      ' COLUMN SEPARATION
 BR1% = 2                                      ' ROW SEPARATION
'-----------------------------------------------------------------
FOR 1% = 1 TO COLB%                            '
BC% = BC1% * 1% + TLCB% - 4                    '  CALCULATE LOCATIONS
FOR i% = 1 TO ROWA%                            '
AR% = BR1% * i% + TLRA% - 1                    '  FOR ARRAYS
FOR j% = 1 TO ACBR%                            '
AC% = BC1% * j% + TLCA% - 4                    '
BR% = BR1% * j% + TLRB% - 1                    '''''''
'-----------------------------------------------------------------
LOCATE AR%, AC%                                '  PRINT
PRINT A%(i%, j%)                               '  ARRAYS
LOCATE BR%, BC%                                '
```

```
PRINT B%(j%, 1%)                                   '''''''''
NEXT j%                                            '  LOOP
NEXT i%                                            '
NEXT 1%                                            ''''''''''
DO
LOOP WHILE INKEY$ = ""

'------------------------------
CALL BLINK(A%(), B%(), MC%, RC%, ROWA%, COLB%, COLA%)
'------------------------------

COLOR 1, 7                                         ''''''''''''''''''''
LOCATE 22, 1                                       ' Erase
PRINT "                   "                        ' Words
PRINT "                   "                        '
PRINT "                   ";                       ''''''''''''''''''''

'--------------------------
'  CONTINUE OR QUIT SEQUENCE
'--------------------------
CALL box(8, 3, 27, 1, 15, 0, "y", 7, 1, "y")
COLOR 11, 0
LOCATE 9, 7
PRINT "(A)gain - (M)enu - (Q)uit";
COLOR 14, 0
LOCATE 9, 7: PRINT "(A)";
LOCATE 9, 17: PRINT "(M)";
LOCATE 9, 26: PRINT "(Q)";
111 DO
sel$ = INKEY$
LOOP WHILE sel$ = ""
IF sel$ = "A" OR sel$ = "a" THEN GOTO 1
IF sel$ = "M" OR sel$ = "m" THEN GOTO 777
IF sel$ = "Q" OR sel$ = "q" THEN GOTO 666
GOTO 111

'==================================
COFACTOR:
'==================================

205

 name$ = "cofactor"
cb% = 7                ' COLOR OF BACKGROUND
COLOR 15, cb%
CLS

'-----------------------------------------
 item$(1) = "     C O F A C T O R    M A T R I X   "
 CALL box(1, 20, 39, 1, 15, 4, "n", 0, 1, "y")
'-----------------------------------------
```

```
  TLRA% = 6                                 'Top Left Row A MATRIX
  TLCA% = 6                                'Top Left Column A
MATRIX
  l% = 13                                   'length of 3x3 box
  o% = 3
'-------------------------
' LOCATION OF MATRIX BOXES
'-------------------------

 CALL box(TLRA%, TLCA%, l%, o%, 15, 1, "y", 0, 2, "y")_
'matrix A

 CALL box(TLRA%, TLCA% + 50, l%, o%, 15, 2, "y", 0, 2,_
"y")'cofactor matrix

 COLOR 7, 0
'''''''''''''''''''''''
 LOCATE TLRA% + 7, TLCA% + 8                 '
 PRINT "[A]"                                 'matrix labels
 LOCATE TLRA% + 7, TLCA% + 50                '
                                             '
 PRINT "Cofactor Matrix [A]"                 '

'------------------------------------------------------

IF sel2$ = "c" GOTO 876
IF sel2$ = "a" GOTO 876

FOR i% = 1 TO o%
''''''''''''''''''''''''''''''''''''''''''''''''''''''
FOR j% = 1 TO o%                 '
RANDOMIZE TIMER                  ' pick random numbers for array
A
pick% = INT(18 * RND) - 9        '  between -9 and 9
A%(i%, j%) = pick%               '
NEXT j%                                 '
NEXT i%
''''''''''''''''''''''''''''''''''''''''''''''''''''''
'calculate the determinant
'---------------------------
deta% = 0
IS1% = A%(1, 1) * 1 * (A%(2, 2) * A%(3, 3) - A%(3, 2) *
A%(2, 3))
IS2% = A%(1, 2) * -1 * (A%(2, 1) * A%(3, 3) - A%(3, 1) *
A%(2, 3))
IS3% = A%(1, 3) * 1 * (A%(2, 1) * A%(3, 2) - A%(3, 1) *
A%(2, 2))
deta% = IS1% + IS2% + IS3%
'----------------------------

876
```

```
'calculate the determinant
'---------------------------
deta% = 0
IS1% = A%(1, 1) * 1 * (A%(2, 2) * A%(3, 3) - A%(3, 2) *
A%(2, 3))
IS2% = A%(1, 2) * -1 * (A%(2, 1) * A%(3, 3) - A%(3, 1) *
A%(2, 3))
IS3% = A%(1, 3) * 1 * (A%(2, 1) * A%(3, 2) - A%(3, 1) *
A%(2, 2))
deta% = IS1% + IS2% + IS3%
'LOCATE 5, 35
'COLOR 0, 7
'PRINT "|A| = "; deta%
'----------------------------

                                       ''''''''''''''''''''''
 BC1% = 6                              ' COLUMN SEPARATION
 BR1% = 2                              ' ROW SEPARATION
                                       '
FOR i% = 1 TO o%
''''''''''''''''''''''''''''''
AR% = BR1% * i% + TLRA% - 1            '
FOR j% = 1 TO o%                       '
AC% = BC1% * j% + TLCA% - 4            '
                                       ''''''''''''''''''''''
COLOR 15, 1                            ' MATRIX A COLOR
                                       ''''''''''''''''''''''
LOCATE AR%, AC%                        '  PRINT
PRINT A%(i%, j%)                       '  ARRAY
NEXT j%                                '
NEXT i%                                '
                   ''''''''''''''''''''''''''''''''''''''
hfc% = 0           ' Foreground color of highlighted rows_
and coloumns
hbc% = 7              ' Background color of highlighted
rows and coloumns
                      '
IF sel$ = "3" THEN    '
                      '''''''''''''''''''''''''''''
ijr% = 8              ' i and j box row location
ijc% = 34             ' i and j box column location
ijfc% = 0             ' i and j foreground color
ijbc% = 7             ' i and j background color
                      '
ELSE                  '''''''''''''''''''''''''''''
                      '
ijr% = 6              ' location of i and j
ijc% = 64             '      box
ijfc% = 0             ' foreground color of i and j
ijbc% = 7             ' background color of i and j
                      '''''''''''''''''''''''''''''
END IF
```

```
i% = 1
j% = 1

'*******************************************************

'-------------------------------------------------------
CALL box(22, 13, 50, 1, 15, 0, "y", cb%, 1, "y")'equation
box
COLOR 11, 0
LOCATE 23, 21
PRINT "Hit the <Space Bar> to watch the process"
COLOR 14, 0
LOCATE 23, 29
PRINT "<Space Bar>"
'-----------------------------------------------------------

DO
LOOP UNTIL INKEY$ <> ""

 CALL box(ijr%, ijc%, 8, 2, ijfc%, ijbc%, "y", 7, 1, "y")_
'print i and j box
 CALL iandj(i%, j%, hfc%, hbc%, ijr%, ijc%, ijfc%, ijbc%)_
'PRINT i AND j

 COLOR hfc%, hbc%
 C% = (TLCA% + 2) + ((j% - 1) * 6)
 R% = (TLRA% + 1) + ((i% - 1) * 2)
 LOCATE R%, C%

 GOSUB rowfirst
 GOSUB colfirst

 COLOR 14, 4                   'color of row column intersection
 LOCATE R%, C%
 PRINT A%(i%, j%)

IF o% = 3 THEN

 CALL minor(i%, j%, A%(), name$, sol%, tsol%)

END IF

 LOCATE R%, C% + 50
 PRINT "███"                   'print white solution box

''''''''''''''''''''''''''
' the  loop starts here
''''''''''''''''''''''''''

FOR l% = 1 TO 3

FOR m% = 1 TO 3
```

```
    DO
    LOOP UNTIL INKEY$ <> ""

   LOCATE R%, C% + 50
   COLOR 15, 2
   PRINT sol%                  'print solution in cofactor matrix

   C%(1%, m%) = sol%           'fill the C matrix with cofactors

       GOSUB rt

   LOCATE R%, C% + 50
   PRINT "███"                    'print white solution box

  NEXT m%

   LOCATE R%, C% + 50
   COLOR 15, 2
   PRINT sol%                    'print solution in cofactor
  matrix

       GOSUB dn

   LOCATE R%, C% + 50
   PRINT "███"                    'print solution box in cofactor
  matrix

  NEXT 1%

  RETURN
  END

  '=======================================================
  dn:                              ' move row down
  '========================================================
   C% = (TLCA% + 2)
   R% = (TLRA% + 1) + ((i% - 1) * 2)
  '------------------------
  ' make row white
  '------------------------
   COLOR 15, 1
   LOCATE R%, C%
   GOSUB rowfirst
  '------------------------
  ' make row yellow
  '------------------------

   i% = i% + 1

   IF i% > o% THEN

  '+++++++++++++++++++++++++++++++++++++++++++
' THE COFACTOR
' PROCESS IS FINISHED
'++++++++++++++++++++++++++++++++++++++++++

R% = R% - 4

    LOCATE R%, C%: PRINT A%(1, 1)
''''''''''''''''''
    LOCATE R% + 1, C%: PRINT "   "              '
    LOCATE R% + 2, C%: PRINT A%(2, 1)           ' blue column
    LOCATE R% + 3, C%: PRINT "   "              '
    LOCATE R% + 4, C%: PRINT A%(3, 1)
''''''''''''''''''
                                                      '''''''
 CALL box(ijr%, ijc%, 8, 2, 7, 7, "y", 7, 1, "y")  'clear i
                                                      'and j
box
 CALL clr1(16, 20, 1, 1)                              'clear
                                                      'minor
box

 TR% = 18
 TC% = 13

 CALL box(TR%, TC%, 50, 3, 0, 7, "y", 7, 1, "n")

 LOCATE TR% + 1, TC% + 4
 PRINT "  The ADJOINT MATRIX is created by TRANSPOSING"
 LOCATE TR% + 2, TC% + 4
 PRINT "  the COFACTOR MATRIX. "
 LOCATE TR% + 4, TC% + 4
 PRINT "  Hit <C>ontinue to create the ADJOINT MATRIX  "
 LOCATE TR% + 5, TC% + 4
 PRINT "  or  (S)top for other selections."
 LOCATE TR% + 4, TC% + 10
 COLOR 14, 0
 PRINT ; "(C)"; : COLOR 11, 0: PRINT "ontinue"
 LOCATE TR% + 5, TC% + 10
 COLOR 14, 0
 PRINT ; "(S)"; : COLOR 11, 0: PRINT "top"

543
DO
sel3$ = INKEY$
LOOP UNTIL sel3$ <> ""

IF sel3$ <> "s" AND sel3$ <> "S" AND sel3$ <> "c" AND sel3$_
<> "C" GOTO 543

IF sel3$ = "s" OR sel3$ = "S" THEN
CALL clr2(TR%, TC%, TR% + 6, TC% + 57)
CALL box(22, 13, 50, 1, 15, 0, "y", cb%, 1, "y")'equation
box
GOSUB escape

ELSEIF sel3$ = "c" OR sel3$ = "C" THEN
CALL clr2(TR%, TC%, TR% + 6, TC% + 57)

''''''''''''''''''''''''''''''''''''''
' PERFORM THE ADJOINT SEQUENCE
''''''''''''''''''''''''''''''''''''''

''''''' '''''''''''''''''''''''''''''
'MOVE the COFACTOR MATRIX TO THE LEFT
''''''''''''''''''''''''''''''''''''''

 CALL clr2(TLRA%, TLCA% - 1, TLRA% + 7, TLCA% + 25) 'ERASE
MATRIX [A]

FOR tlcc% = 50 TO 6 STEP -2

CALL box(6, tlcc%, 15, o%, 15, 2, "y", 0, 2, "y")'cofactor_
matrix

   COLOR 7, 0
'''''''''''''''''''''''
   LOCATE 13, tlcc%          '
   PRINT ; "Cofactor Matrix [A] "; : COLOR 7, 7: PRINT "    "

  FOR p% = 1 TO o%                               ''''''''''''''''''''''
   AR% = BR1% * p% + TLRA% - 1                   '
  FOR q% = 1 TO o%                               '
   AC% = BC1% * q% + tlcc% - 4                   '
                                                 ''''''''''''''''''''''
   COLOR 15, 2                                   ' MATRIX C COLOR
                                                 ''''''''''''''''''''''
   LOCATE AR%, AC%                               '  PRINT
   PRINT C%(p%, q%)                              '  ARRAY
  NEXT q%                                        '
  NEXT p%                                        '

   CALL clr2(6, tlcc% + 21, 13, tlcc% + 27)

FOR jj% = 1 TO 2000
NEXT jj%

NEXT tlcc%

'-----------------------------------------------------
 item$(1) = "      A D J O I N T     M A T R I X   "
 CALL box(1, 20, 39, 1, 15, 1, "n", 0, 1, "y")
'-----------------------------------------------------
```

```
END IF

 TLCB% = 55                'Top Left Column B MATRIX
 TLRB% = 6                'Top Left Column B MATRIX

CALL box(TLRB%, TLCB%, 15, 3, 15, 4, "y", 0, 2, "y")'matrix_
AT
 LOCATE TLRA% + 7, TLCA% + 49
COLOR 7, 0

 PRINT " Adjoint Matrix [A]"


'---------------------------------------------------------
cb% = 7                                         'BACKGROUND COLOR
CALL box(22, 13, 50, 1, 15, 0, "y", cb%, 1, "y")_
'equation box
COLOR 11, 0
LOCATE 23, 21
PRINT "Hit the <Space Bar> to watch the process"
COLOR 14, 0
LOCATE 23, 29
PRINT "<Space Bar>"
'---------------------------------------------------
COLOR 15, 2

DO
LOOP WHILE INKEY$ = ""

' Perform the transposition

RC$ = "r"    'THIS IS HARDWIRED  FOR A ROW EXCHANGE

IF RC$ = "r" OR RC$ = "R" THEN
BLNKC% = 14                                  ' BLINK COLOR

FOR i% = 1 TO 3
  AR% = BR1% * i% + TLRA% - 1              '  FOR ARRAY
  BC% = TLCB% + 2 + ((i% - 1) * 6)

FOR j% = 1 TO 3
  AC% = BC1% * j% + TLCA% - 4
  BR% = BR1% * j% + TLRB% - 1

LOCATE AR%, AC%                              '  BLINK
COLOR 30                                     '  THE
PRINT C%(i%, j%)                             '  ARRAY ELEMENT

D%(j%, i%) = C%(i%, j%)   '******* MAKE D THE INVERSE

  COLOR 15
  LOCATE BR%, BC%                             ' PRINT
  PRINT "█"                                         ' RESULT BOX

DO
LOOP WHILE INKEY$ = ""
COLOR 15, 2                                        '  TURN OFF BLINK
LOCATE AR%, AC%                                    '  PRINT
PRINT C%(i%, j%)                                   '  NON BLINKING ARRAY
ELEMENT
  COLOR 15, 4
  LOCATE BR%, BC%                                  ' PRINT
  PRINT C%(i%, j%)                                 ' TRANSPOSED ELEMENT
  COLOR 15, 2
NEXT j%
NEXT i%

ELSEIF RC$ = "c" OR RC$ = "C" THEN

BLNKC% = 14                                        ' BLINK COLOR
FOR j% = 1 TO 3                                    '
 AC% = BC1% * j% + TLCA% - 4                       '
 BR% = BR1% * j% + TLRB% - 1                       '

FOR i% = 1 TO 3                                    '
 AR% = BR1% * i% + TLRA% - 1                       '  FOR ARRAY
 BC% = TLCB% + 2 + ((i% - 1) * 6)

LOCATE AR%, AC%                                    '  BLINK
COLOR 30                                           '  THE
PRINT C%(i%, j%)                                   '  ARRAY ELEMENT

  COLOR 15
  LOCATE BR%, BC%                                   ' PRINT
  PRINT "█"                                         ' RESULT BOX

DO
LOOP WHILE INKEY$ = ""

COLOR 15, 2                                        '  TURN OFF BLINK
LOCATE AR%, AC%                                    '  PRINT
PRINT C%(i%, j%)                                   '  NON BLINKING ARRAY
ELEMENT
  COLOR 15, 4
  LOCATE BR%, BC%                                  ' PRINT
  PRINT C%(i%, j%)                                 ' TRANSPOSED ELEMENT
  COLOR 15, 2
NEXT i%
NEXT j%

 END IF

'==========================================================
 TR% = 18
 TC% = 13


 CALL box(TR%, TC%, 50, 3, 0, 7, "y", 7, 1, "n")

 LOCATE TR% + 1, TC% + 4
 PRINT "  The INVERTED MATRIX is created by dividing"
 LOCATE TR% + 2, TC% + 4
 PRINT "  the ADJOINT MATRIX by the MATRIX DETERMINANT. "
 LOCATE TR% + 4, TC% + 4
 PRINT "  Hit <C>ontinue to INVERT the MATRIX  "
 LOCATE TR% + 5, TC% + 4
 PRINT "  or  (S)top for other selections."
 LOCATE TR% + 4, TC% + 10
 COLOR 14, 0
 PRINT ; "(C)"; : COLOR 11, 0: PRINT "ontinue"
 LOCATE TR% + 5, TC% + 10
 COLOR 14, 0
 PRINT ; "(S)"; : COLOR 11, 0: PRINT "top"

544
DO
sel3$ = INKEY$
LOOP UNTIL sel3$ <> ""

IF sel3$ <> "s" AND sel3$ <> "S" AND sel3$ <> "c" AND sel3$
<> "C" GOTO 544

IF sel3$ = "s" OR sel3$ = "S" THEN
CALL clr2(TR%, TC%, TR% + 6, TC% + 57)
CALL box(22, 13, 50, 1, 15, 0, "y", cb%, 1, "y")_
'equation box
GOSUB escape

ELSEIF sel3$ = "c" OR sel3$ = "C" THEN
CALL clr2(TR%, TC%, TR% + 6, TC% + 57)

''''''''''''''''''''''''''''''''''''''
' PERFORM THE INVERSION SEQUENCE
''''''''''''''''''''''''''''''''''''''
'This is where the adjoint should be moved, and
'the derterminent calculated so that it can
'be divided into the adjoint.
''''''''''''''''''''''''''''''''''''''
END IF

''''''''''''''''''''''''''''''''''''''
'MOVE the adjoint MATRIX TO THE LEFT
''''''''''''''''''''''''''''''''''''''
```

```
 CALL clr2(TLRA%, TLCA% - 1, TLRA% + 7, TLCA% + 25) 'ERASE
MATRIX [A]

FOR tlcc% = 50 TO 10 STEP -2

CALL box(6, tlcc%, 15, o%, 15, 4, "y", 0, 2, "y")'cofactor_
matrix

   COLOR 7, 0
''''''''''''''''''''''''
   LOCATE 13, tlcc%                                      '
   PRINT ; "Adjoint Matrix [A] "; : COLOR 7, 7: PRINT "    "

  FOR p% = 1 TO o%                                      ''''''''''''''''''''''
   AR% = BR1% * p% + TLRA% - 1                          '
  FOR q% = 1 TO o%                                      '
   AC% = BC1% * q% + tlcc% - 4                          '
                                                        '''''''''''''''''''''''
   COLOR 15, 4                                          ' MATRIX C COLOR
                                                        '''''''''''''''''''''''
   LOCATE AR%, AC%                                      '  PRINT
   PRINT D%(p%, q%)                                     '  ARRAY
  NEXT q%                                               '
  NEXT p%                                               '

   CALL clr2(6, tlcc% + 21, 13, tlcc% + 27)

FOR jj% = 1 TO 2000
NEXT jj%

NEXT tlcc%

'-----------------------------------------------------
 item$(1) = "T H E   I N V E R T E D   M A T R I X"
 CALL box(1, 20, 39, 1, 15, 2, "n", 0, 1, "y")
'-----------------------------------------------------
LOCATE 5, 35
COLOR 0, 7
PRINT "|A| = "; deta%

'calculate the determinant
'--------------------------
totsum% = 0
IS1% = A%(1, 1) * 1 * (A%(2, 2) * A%(3, 3) - A%(3, 2) *
A%(2, 3))
IS2% = A%(1, 2) * -1 * (A%(2, 1) * A%(3, 3) - A%(3, 1) *
A%(2, 3))
IS3% = A%(1, 3) * 1 * (A%(2, 1) * A%(3, 2) - A%(3, 1) *
A%(2, 2))
totsum% = IS1% + IS2% + IS3%
'---------------------------
```

```
LOCATE 9, 38
COLOR 15, 7: PRINT ; "÷"; SPC(4);
LOCATE 9, 50
COLOR 15, 7: PRINT ; "=";

TLRB% = 6
TLCB% = 55

CALL box(TLRB%, TLCB%, 15, o%, 15, 1, "y", 0, 2, "y")
'inverted matrix box
   COLOR 7, 0
   LOCATE 13, TLCB%
   PRINT ; "Inverted Matrix [A] ";

CALL box(22, 13, 50, 1, 15, 0, "y", cb%, 1, "y")
'equation box
COLOR 11, 0
LOCATE 23, 21
PRINT "Hit the <Space Bar> to watch the process"
COLOR 14, 0
LOCATE 23, 29
PRINT "<Space Bar>"

                                       '''''''''''''''''''''''''
 BC1% = 6                              ' COLUMN SEPARATION
 BR1% = 2                              ' ROW SEPARATION
 TLRA% = 6                             'Top Left Row A MATRIX
 TLCA% = 10                            'Top Left Column A MATRIX
                                       '
                                       '''''''''''''''''''''''''
'''''''''''''''''''''
' locate the scalar
'''''''''''''''''''''
sc% = 0                  'scalar color

COLOR sc%, cb%
LOCATE 9, 40: PRINT "|A|"
totsum%
COLOR 15, 1

DO
LOOP UNTIL INKEY$ <> ""

COLOR 30, cb%
 LOCATE 9, 40: PRINT "|A|"                              'totsum%

IF totsum% = 0 THEN
 TR% = 18
 TC% = 13
 CALL box(TR%, TC%, 50, 3, 0, 7, "y", 7, 1, "n")
 LOCATE TR% + 1, TC% + 4
 PRINT "  The DETERMINANT of Matrix A is 0.      "
 LOCATE TR% + 2, TC% + 4
 PRINT "  You cannot divide by 0."
 LOCATE TR% + 4, TC% + 4
 PRINT "  Hit any key to return to the INVERSION MENU.  "

 DO
 LOOP UNTIL INKEY$ <> ""
 GOTO 98

END IF

COLOR 15, 4
BLNKC% = 14                                   ' BLINK COLOR
                                              '
FOR i% = 1 TO 3                               '
AR% = BR1% * i% + TLRA% - 1                   '
FOR j% = 1 TO 3                               '
AC% = BC1% * j% + TLCA% - 4                   '
                                              '
BR% = AR%                                     '
BC% = BC1% * j% + TLCB% - 4                   '
                                              '
'-----------------------------------------------------

COLOR 30, 4

LOCATE AR%, AC%                               '  BLINK
PRINT D%(i%, j%)                              '  D ARRAY ELEMENTS

COLOR 15, 1
RC% = BC1% * j% + TLCB% - 4                   '
  LOCATE BR%, RC%                             ' PRINT
  PRINT "█"                                   ' RESULT BOX

S! = D%(i%, j%) / totsum%
E!(i%, j%) = D%(i%, j%) / totsum%

'------------------------------------------------------
COLOR 11, 0          ' clear equation box at bottom
LOCATE 23, 40        ' of screen
PRINT SPC(24);
'-------------------------------------------------------

   COLOR 15, 0
   LOCATE 23, 45
''''''''''
PRINT USING "(####  ÷ ####) =  █"; D%(i%, j%); totsum%;
'PRINT
DO
LOOP WHILE INKEY$ = ""

COLOR 15, 4                                   '  TURN OFF BLINK
```

```
LOCATE AR%, AC%                                  '  PRINT
PRINT D%(i%, j%)                                 '  NON BLINKING D ARRAY
ELEMENT

  COLOR 15, 1
  LOCATE BR%, RC%                                ' PRINT
IF SEL1$ = "3" THEN
  PRINT S%                                       ' result
ELSE

  PRINT USING "##.##"; E!(i%, j%)

' PRINT USING "##.##"; S!

 END IF

COLOR 15, 1

NEXT j%
NEXT i%

'''''''''''''''''''''''''
'turn off blinking scalar
'
'''''''''''''''''''''''''
COLOR 0, 7
 LOCATE 9, 40: PRINT "|A|"                               'totsum%

'==================
GOSUB cescape
'===================

DO
LOOP UNTIL INKEY$ <> ""
GOTO 666

'=====================================================

 GOSUB escape

 END IF

CALL iandj(i%, j%, hfc%, hbc%, ijr%, ijc%, ijfc%, ijbc%)

  R% = (TLRA% + 1) + ((i% - 1) * 2)
  C% = (TLCA% + 2)

 LOCATE R%, C%
 GOSUB rowfirst

C% = (TLCA% + 2) + ((j% - 1) * 6)
R% = TLRA% + 1
```

```
GOSUB colfirst

 R% = (TLRA% + 1) + ((i% - 1) * 2)
 C% = (TLCA% + 2) + ((j% - 1) * 6)
                                                  ''''''''''''''''''
COLOR 14, 4                                       'yellow on red
LOCATE R%, C%                                     '
PRINT A%(i%, j%)                                  '
                                                  ''''''''''''''''''

COLOR hfc%, hbc%
IF o% = 3 THEN
''''''''''''''''''
 CALL minor(i%, j%, A%(), name$, sol%, tsol%)   'print minor
END IF
''''''''''''''''''
RETURN
END
'=======================================================
rt:                            ' move column RIGHT
'========================================================

 R% = (TLRA% + 1)
 C% = (TLCA% + 2) + ((j% - 1) * 6)
COLOR 15, 1

'-----------------------
' make column white
'-----------------------

GOSUB colfirst

'----------------------
'make next column yellow
'----------------------

j% = j% + 1
IF j% > o% THEN j% = 1

COLOR hfc%, hbc%

  C% = (TLCA% + 2) + ((j% - 1) * 6)
  R% = (TLRA% + 1)

GOSUB colfirst

CALL iandj(i%, j%, hfc%, hbc%, ijr%, ijc%, ijfc%, ijbc%)
'i AND j

R% = (TLRA% + 1) + ((i% - 1) * 2)
LOCATE R%, (TLCA% + 2)
```

```
 GOSUB rowfirst

 R% = (TLRA% + 1) + ((i% - 1) * 2)
 C% = (TLCA% + 2) + ((j% - 1) * 6)

COLOR 14, 4
LOCATE R%, C%
PRINT A%(i%, j%)
COLOR hfc%, hbc%

IF o% = 3 THEN
 CALL minor(i%, j%, A%(), name$, sol%, tsol%)
END IF
RETURN
END

'==================================================================
determin:
'==================================================================
tsol% = 0
ijflag$ = "0"
'****   ORDER OF THE MATRIX   ****
IF o% = 3 THEN
'*********************************
  l% = 13                               'length of 3x3 box
 TLRA% = 6                             'Top Left Row A MATRIX
 TLCA% = 3                            'Top Left Column A MATRIX
ELSEIF o% = 2 THEN
'*********************************
  l% = 7                                'length of 2x2 box
  TLRA% = 11                            'Top Left Row A MATRIX
  TLCA% = 5                            'Top Left Column A MATRIX
END IF

'----------------------------------------------
cb% = 7                        ' COLOR OF BACKGROUND
COLOR 15, cb%
CLS
'---------------------------------------------------
 item$(1) = "  M A T R I X   D E T E R M I N A N T"
 CALL box(1, 20, 39, 1, 15, 2, "n", 0, 1, "y")
'---------------------------------------------------
 R1% = 7
 C1% = 10

IF o% = 2 AND sel2$ <> "a" THEN

 CALL box(R1% - 1, C1% - 2, 62, 4, 15, 1, "y", 0, 1, "y")
 LOCATE R1%, C1%
 PRINT " The DETERMINANT can only be calculated for SQUARE_
matrices."
 LOCATE R1% + 1, C1%
 PRINT " Since a SQUARE matrix has the same number of ROWS_
and COLUMNS,"
 LOCATE R1% + 2, C1%
 PRINT " the size of the matrix is expressed using a single_
term known"
 LOCATE R1% + 3, C1%
 PRINT " as the ORDER of the matrix.  "
 LOCATE R1% + 5, C1%
 PRINT " Hit any key to see how to find the DETERMINANT of
a_ matrix of"
 LOCATE R1% + 6, C1%
 PRINT " ORDER 2."

DO
LOOP UNTIL INKEY$ <> ""
ELSE
END IF

                                 ''''''''''''''''''''''''
COLOR 7, 7
FOR n% = 1 TO 10                 '  clear explanation
LOCATE R1% + n% - 2, C1% - 3     '   box
PRINT SPC(80 - C1%);             '
NEXT n%                          ''''''''''''''''''''''''

'------------------------
' LOCATION OF MATRIX BOX
'------------------------
CALL box(TLRA%, TLCA%, 1%, o%, 15, 1, "y", 0, 3, "y")_
'matrix A

IF o% = 2 THEN
''''''''''''''''''''''''''''''''''''
 COLOR 15, 7                       '  place equal sign for
 LOCATE TLRA% + 2, TLCA% + 15      '
 PRINT "="                         '  an order 2 matrix
END IF                             '
                                   '

IF sel2$ = "A" OR sel2$ = "a" THEN GOTO 1212

 FOR i% = 1 TO o%
''''''''''''''''''''''''''''''''''''
                                    '
 FOR j% = 1 TO o%                   '
  RANDOMIZE TIMER                   '  pick random numbers
                                    '  for array A
  pick% = INT(18 * RND) - 9         '  between -9 and 9
  A%(i%, j%) = pick%                '
 NEXT j%                            '
 NEXT i%
''''''''''''''''''''''''''''''''''''


'Make 1 in 5 singular
```

```
  RANDOMIZE TIMER          '  pick random numbers for array
A
  sing% = INT(3 * RND)     '  between 0 and 2
IF sing% = 2 THEN
FOR j% = 1 TO o%
A%(1, j%) = A%(2, j%)
NEXT j%
END IF

'calculate the determinant
'----------------------------
deta% = 0
IS1% = A%(1, 1) * 1 * (A%(2, 2) * A%(3, 3) - A%(3, 2) *
A%(2, 3))
IS2% = A%(1, 2) * -1 * (A%(2, 1) * A%(3, 3) - A%(3, 1) *
A%(2, 3))
IS3% = A%(1, 3) * 1 * (A%(2, 1) * A%(3, 2) - A%(3, 1) *
A%(2, 2))
deta% = IS1% + IS2% + IS3%
'-----------------------------

1212
                                    '
 BC1% = 6                           ' COLUMN SEPARATION
 BR1% = 2                           ' ROW SEPARATION
FOR i% = 1 TO o%                    '
AR% = BR1% * i% + TLRA% - 1         '
FOR j% = 1 TO o%                    '
AC% = BC1% * j% + TLCA% - 4         '
                                    ''''''''''''''''''
COLOR 15, 1                         ' MATRIX  COLOR
                                    ''''''''''''''''''
LOCATE AR%, AC%                     '  PRINT
PRINT A%(i%, j%)                    '  ARRAY
NEXT j%                             '
NEXT i%                             '

hfc% = 0  ' Foreground color of highlighted rows and
coloumns
hbc% = 7  ' Background color of highlighted rows and
coloumns

'*********************************************************

boxr% = 22
boxc% = 18
'------------------------------------------------------
item$(1) = "EXPAND the MATRIX along a <R>ow or <C>olumn?"
CALL box(boxr%, boxc%, 44, 1, 15, 0, "n", 7, 1, "n")
'------------------------------------------------------
 COLOR 12, 0
LOCATE boxr% + 1, boxc% + 3
```

```
PRINT ; "EXPAND";
 COLOR 15
PRINT " the MATRIX along a "
 COLOR 14, 0
LOCATE boxr% + 1, boxc% + 29
PRINT ; "<R>";
 COLOR 11: PRINT "ow";
LOCATE boxr% + 1, boxc% + 38
 COLOR 14: PRINT ; "<C>";
 COLOR 11: PRINT "olumn";
'------------------------------------------------

77

 DO
 pick$ = INKEY$
 LOOP UNTIL pick$ <> ""

 COLOR hfc%, hbc%
i% = 1
j% = 1
C% = (TLCA% + 2) + ((j% - 1) * 6)
R% = (TLRA% + 1) + ((i% - 1) * 2)
                            ''''''''''''''''''''''''''''
ijr% = 1                    ' location of i and j
ijc% = 4                    '     box
ijfc% = 0                   ' foreground color of i and j
ijbc% = 7                   ' background color of i and j
                            ''''''''''''''''''''''''''''
'==================================================
'                  ROW IS PICKED FIRST
'==================================================

 IF pick$ = "r" OR pick$ = "R" THEN
'-----------------------------------------------
CALL box(boxr%, boxc%, 44, 1, 7, 7, "y", 7, 1, "n")
'-----------------------------------------------
 COLOR hfc%, hbc%
    name$ = "ROW"
    flag$ = "0"

    LOCATE R%, C%

 GOSUB rowfirst
 GOSUB abox

24       'return here

KEY 16, CHR$(&HO) + CHR$(57)         ' <Space> - ?
KEY 15, CHR$(&HO) + CHR$(1)          ' <Esc>   - QUIT
KEY 17, CHR$(&HO) + CHR$(28)         ' <Enter> - ?
```

```
        KEY(15) ON
        KEY(16) ON
        KEY(17) ON
29
    ON KEY(16) GOSUB MOVEDN
    ON KEY(15) GOSUB EXT
    ON KEY(17) GOSUB rtn

GOTO 29

    GOTO 88

'=================================================
'               COLUMN IS PICKED FIRST
'=================================================

  ELSEIF pick$ = "C" OR pick$ = "c" THEN
    name$ = "COLUMN"
    flag$ = "1"
'----------------------------------------------
CALL box(boxr%, boxc%, 44, 1, 7, 7, "y", 7, 1, "n")
'----------------------------------------------
 COLOR hfc%, hbc%

 GOSUB colfirst
 GOSUB abox

25       'return here

KEY 17, CHR$(&H0) + CHR$(28)         'enter - ?
KEY 16, CHR$(&H0) + CHR$(57)         ' <Space> - ?
KEY 15, CHR$(&H0) + CHR$(1)         'esc - QUIT
       KEY(15) ON
       KEY(16) ON
       KEY(17) ON
26
    ON KEY(15) GOSUB EXT
    ON KEY(16) GOSUB MOVERT
    ON KEY(17) GOSUB rtn
GOTO 26
    GOTO 88
    ELSE GOTO 77
88  END IF
'------------------------------------------
133
'----------------------------------------------
boxr% = 22
boxc% = 10
item$(1) = "      Hit <Space Bar> to EXPAND the MATRIX"
CALL box(boxr%, boxc%, 54, 1, 15, 0, "n", 7, 1, "n")
'----------------------------------------------
 COLOR 14, 0
 LOCATE boxr% + 1, boxc% + 13
 PRINT "<Space Bar>";
'----------------------------------------------

'------------------------------------
' COLUMN POSITION after row selection
'------------------------------------
 IF name$ = "ROW" THEN
   C% = (TLCA% + 2) + ((j% - 1) * 6)
   R% = (TLRA% + 1)
    COLOR hfc%, hbc%

GOSUB colfirst

 R% = (TLRA% + 1) + ((i% - 1) * 2)
 COLOR 14, 4                'color of row column intersection
 LOCATE R%, C%
 PRINT A%(i%, j%)

'------------------------------------
' ROW POSITION after column selection
'------------------------------------

  ELSEIF name$ = "COLUMN" THEN
    C% = (TLCA% + 2)
    R% = (TLRA% + 1)
    COLOR hfc%, hbc%
    LOCATE R%, C%

 GOSUB rowfirst

 C% = (TLCA% + 2) + ((j% - 1) * 6)
 COLOR 14, 4              'color of row column intersection
 LOCATE R%, C%
 PRINT A%(i%, j%)

 END IF

COLOR hfc%, hbc%
IF o% = 3 THEN
 CALL minor(i%, j%, A%(), name$, sol%, tsol%)
ELSEIF o% = 2 THEN
 CALL determ2(i%, j%, A%(), name$, sol%)
END IF


101       'return here

IF o% = 3 THEN
IF name$ = "COLUMN" AND i% = 3 THEN GOSUB nescape
IF name$ = "ROW" AND j% = 3 THEN GOSUB nescape
END IF
```

```
IF o% = 2 THEN
IF name$ = "COLUMN" AND i% = 2 THEN GOSUB nescape
IF name$ = "ROW" AND j% = 2 THEN GOSUB nescape
END IF

DO
LOOP UNTIL INKEY$ <> ""

IF name$ = "COLUMN" THEN
GOSUB MOVEDN

ELSEIF name$ = "ROW" THEN
GOSUB MOVERT
END IF

'=========================================================
MOVEDN:                              ' move determinent row down
'========================================================
 C% = (TLCA% + 2)
 R% = (TLRA% + 1) + ((i% - 1) * 2)

 KEY(11) OFF
 KEY(12) OFF
 KEY(13) OFF
 KEY(14) OFF
 KEY(15) OFF
 KEY(16) OFF
 KEY(17) OFF
'------------------------
' make row white
'------------------------
 COLOR 15, 1
 LOCATE R%, C%
 GOSUB rowfirst
'------------------------
' make row yellow
'------------------------

 i% = i% + 1
 IF i% > o% THEN i% = 1
IF ijflag$ = "1" THEN
 CALL iandj(i%, j%, hfc%, hbc%, ijr%, ijc%, ijfc%, ijbc%)
END IF
 COLOR hfc%, hbc%

  R% = (TLRA% + 1) + ((i% - 1) * 2)
  C% = (TLCA% + 2)

 LOCATE R%, C%
 GOSUB rowfirst

IF flag$ = "0" GOTO 24
```

```
C% = (TLCA% + 2) + ((j% - 1) * 6)
R% = TLRA% + 1

GOSUB colfirst

 R% = (TLRA% + 1) + ((i% - 1) * 2)
 C% = (TLCA% + 2) + ((j% - 1) * 6)

COLOR 14, 4                'yellow on red
LOCATE R%, C%
PRINT A%(i%, j%)


COLOR hfc%, hbc%

IF o% = 3 THEN
 CALL minor(i%, j%, A%(), name$, sol%, tsol%)
ELSEIF o% = 2 THEN
 CALL determ2(i%, j%, A%(), name$, sol%)
END IF

GOTO 101
RETURN
END

'=====================================================
MOVERT:                  ' move determinent column RIGHT
'====================================================

 R% = (TLRA% + 1)
 C% = (TLCA% + 2) + ((j% - 1) * 6)

 KEY(11) OFF
 KEY(12) OFF
 KEY(13) OFF
 KEY(14) OFF
 KEY(15) OFF
 KEY(16) OFF
 KEY(17) OFF

COLOR 15, 1

'-----------------------
' make column white
'-----------------------

GOSUB colfirst

'----------------------
'make next column yellow
'----------------------
```

```
j% = j% + 1
'IF J% >= o% THEN J% = o%
IF j% > o% THEN j% = 1

COLOR hfc%, hbc%

  C% = (TLCA% + 2) + ((j% - 1) * 6)
  R% = (TLRA% + 1)

GOSUB colfirst
IF ijflag$ = "1" THEN
 CALL iandj(i%, j%, hfc%, hbc%, ijr%, ijc%, ijfc%, ijbc%)
END IF
 COLOR hfc%, hbc%


IF flag$ = "1" GOTO 25

R% = (TLRA% + 1) + ((i% - 1) * 2)
LOCATE R%, (TLCA% + 2)

 GOSUB rowfirst

 R% = (TLRA% + 1) + ((i% - 1) * 2)
 C% = (TLCA% + 2) + ((j% - 1) * 6)

COLOR 14, 4
LOCATE R%, C%
PRINT A%(i%, j%)
COLOR hfc%, hbc%

IF o% = 3 THEN
 CALL minor(i%, j%, A%(), name$, sol%, tsol%)
ELSEIF o% = 2 THEN
 CALL determ2(i%, j%, A%(), name$, sol%)
END IF

 GOTO 101

RETURN
END


'=========================================
EXT:
'=========================================
KEY(17) OFF
 KEY(16) OFF
COLOR 7, 7
CLS
'----------------------------------------------------
 item$(1) = "    M A T R I X   I N V E R S I O N "
 CALL box(1, 20, 39, 1, 15, 2, "n", 0, 1, "y")
'-----------------------------------------------------

GOTO 98
RETURN
END

'=======================================================
rtn:
'=======================================================
 CALL box(ijr%, ijc%, 8, 2, ijfc%, ijbc%, "y", 7, 1, "n")
'print i and j box
 CALL iandj(i%, j%, hfc%, hbc%, ijr%, ijc%, ijfc%, ijbc%)
'i AND j
 ijflag$ = "1"
 KEY(11) OFF
 KEY(12) OFF
 KEY(13) OFF
 KEY(14) OFF
 KEY(15) OFF
 KEY(16) OFF
 KEY(17) OFF
    item$(1) = ""
    item$(2) = ""
    CALL box(boxr%, boxc%, 38, 2, 7, 7, "n", 7, 1, "y")
GOTO 133

RETURN
END

'===========================
abox:
'===========================
    boxr% = 20
    boxc% = 21
    item$(1) = "Use <Space Bar> to select the"
    item$(2) = "Hit <Enter> after selection"
    CALL box(boxr%, boxc%, 38, 2, 11, 0, "n", 7, 1, "n")
    LOCATE boxr% + 1, boxc% + 34
    COLOR 12
    PRINT ; name$
    LOCATE boxr% + 1, boxc% + 7
    COLOR 14
    PRINT "<Space Bar>"
    LOCATE boxr% + 3, boxc% + 7
    PRINT "<Enter>"

RETURN
END
'===============================================
row:
'=================================================
    IF o% = 2 THEN
    PRINT USING "##    ## "; A%(i%, j%); A%(i%, j%);
    ELSEIF o% = 3 THEN
    PRINT USING "##    ##    ##"; A%(i%, j%); A%(i%, j%);
A%(i%, j%)
    END IF
 RETURN
 END
'================================================
col:
'================================================

IF o% = 3 THEN
    LOCATE R%, C%: PRINT A%(i%, j%)
'''''''''''''''''

    LOCATE R% + 1, C%: PRINT "   "              '
    LOCATE R% + 2, C%: PRINT A%(i%, j%)         ' grey column
    LOCATE R% + 3, C%: PRINT "   "              '
    LOCATE R% + 4, C%: PRINT A%(i%, j%)
''''''''''''''''''

ELSEIF o% = 2 THEN
    LOCATE R%, C%: PRINT A%(i%, j%)
''''''''''''''''

    LOCATE R% + 1, C%: PRINT "   "    '  grey column
    LOCATE R% + 2, C%: PRINT A%(i%, j%)
''''''''''''''''''''

END IF
RETURN
END

'====================================================
rowfirst:
'====================================================

    IF o% = 2 THEN
    PRINT USING "##    ## "; A%(i%, 1); A%(i%, 2);
    ELSEIF o% = 3 THEN
    PRINT USING "##    ##    ##"; A%(i%, 1); A%(i%, 2);
A%(i%, 3)
    END IF
 RETURN
 END
'==========================================================
colfirst:
'==========================================================

IF o% = 3 THEN
    LOCATE R%, C%: PRINT A%(1, j%)
'''''''''''''''''

    LOCATE R% + 1, C%: PRINT "   "             '
    LOCATE R% + 2, C%: PRINT A%(2, j%)         ' grey column
    LOCATE R% + 3, C%: PRINT "   "             '
    LOCATE R% + 4, C%: PRINT A%(3, j%)
''''''''''''''''''
ELSEIF o% = 2 THEN
    LOCATE R%, C%: PRINT A%(1, j%)
''''''''''''''''''
    LOCATE R% + 1, C%: PRINT "    "             '  grey column
    LOCATE R% + 2, C%: PRINT A%(2, j%)
''''''''''''''''''''
END IF
RETURN
END
'============================================================
cescape:        ' escape from final invert screen
'============================================================
'-------------------------------------------------------
COLOR 12, 0          'clear equation box at bottom
LOCATE 23, 16        ' of screen
PRINT SPC(50);
'--------------------------------------------------------


R% = 22
C% = 14
COLOR 11, 0
LOCATE R% + 1, C% + 2
PRINT "(A)gain    -    (V)erify    -    (M)enu    -    (Q)uit";
COLOR 14, 0
LOCATE R% + 1, C% + 2: PRINT "(A)";
LOCATE R% + 1, C% + 16: PRINT "(V)";
LOCATE R% + 1, C% + 31: PRINT "(M)";
LOCATE R% + 1, C% + 44: PRINT "(Q)";
338
DO
sel2$ = INKEY$
LOOP WHILE sel2$ = ""

IF sel2$ = "A" OR sel2$ = "a" THEN

   CALL clr2(4, 1, 24, 80)
   GOSUB determin

ELSEIF sel2$ = "M" OR sel2$ = "m" THEN

   deta% = 0                    'initialize determinant of A
   CALL clr2(22, 1, 25, 79)
   GOTO 98                      ' determinant menu

ELSEIF sel2$ = "Q" OR sel2$ = "q" THEN

  GOTO 666

ELSEIF sel2$ = "V" OR sel2$ = "v" THEN
```

```
  FOR i% = 1 TO 3
  FOR j% = 1 TO 3

  NEXT j%
  NEXT i%

GOSUB MULTIPLY

END IF

GOTO 338

RETURN
END
'============================================================
nescape:
'============================================================
'-------------------------------------------------------
COLOR 12, 0          'clear equation box at bottom
LOCATE 23, 12        ' of screen
PRINT SPC(56);
'--------------------------------------------------------

R% = 22
C% = 12
COLOR 11, 0
LOCATE R% + 1, C% + 2
PRINT "(A)gain   -   (C)ontinue   -   (M)enu   -   (Q)uit";
COLOR 14, 0
LOCATE R% + 1, C% + 2: PRINT "(A)";
LOCATE R% + 1, C% + 16: PRINT "(C)";
LOCATE R% + 1, C% + 33: PRINT "(M)";
LOCATE R% + 1, C% + 46: PRINT "(Q)";
339
DO
sel2$ = INKEY$
LOOP WHILE sel2$ = ""


IF sel2$ = "A" OR sel2$ = "a" THEN

  CALL clr2(4, 1, 24, 80)
   GOSUB determin
 ELSEIF sel2$ = "M" OR sel2$ = "m" THEN
   deta% = 0                     'initialize determinant of A
   CALL clr2(22, 1, 25, 79)
   GOTO 98                       ' determinant menu

 ELSEIF sel2$ = "Q" OR sel2$ = "q" THEN
  GOTO 666

 ELSEIF sel2$ = "C" OR sel2$ = "c" THEN
```

```
  IF sel$ = "1" THEN
  sel2$ = ""
  sel$ = "2"
  o% = 3
  GOSUB determin
  ELSEIF sel$ = "2" THEN
  sel$ = "3"
  GOTO 205

  END IF
END IF

GOTO 339

RETURN
END
'============================================================
escape:
'============================================================
'-------------------------------------------------------
COLOR 11, 0          'clear equation box at bottom
LOCATE 23, 21        ' of screen
PRINT SPC(46);
'--------------------------------------------------------


R% = 22
C% = 22
COLOR 11, 0
LOCATE R% + 1, C% + 2
PRINT "(A)gain   -   (M)enu   -   (Q)uit";
COLOR 14, 0
LOCATE R% + 1, C% + 2: PRINT "(A)";
LOCATE R% + 1, C% + 16: PRINT "(M)";
LOCATE R% + 1, C% + 29: PRINT "(Q)";
333 DO
sel2$ = INKEY$
LOOP WHILE sel2$ = ""

'---------------------------------------------------------
'If (A)gain is chosen, the GOTO statement must be correct
'---------------------------------------------------------

IF sel2$ = "A" OR sel2$ = "a" THEN


  IF SEL1$ = "1" GOTO 200      ' GOTO addition
  IF SEL1$ = "2" GOTO 200      ' GOTO addition
  IF SEL1$ = "3" GOTO 202      ' GOTO scalar multiplication
  IF SEL1$ = "4" GOTO 202      ' GOTO scalar division
  IF SEL1$ = "5" GOTO 33       ' GOTO addition
  IF sel$ = "3" GOTO 205       ' GOTO cofactor
```

```
 ELSEIF sel2$ = "M" OR sel2$ = "m" THEN

  IF sel$ = "3" GOTO 98                            ' determinant
menu
  GOTO 777

 ELSEIF sel2$ = "Q" OR sel2$ = "q" THEN
  GOTO 666

END IF

GOTO 333

RETURN
END
'============================================================
'            THIS IS THE END
'============================================================
6666
666
   R1% = 1
   C1% = 1
   R2% = 24
   C2% = 80
   klr% = 0
LOCATE R1%, C1%

123
  COLOR klr%
  FOR ii% = C1% TO C2%         ' i% is a column variable
  LOCATE R1%, ii%
  PRINT ; "█";

dt% = 10                       ' delay time

    FOR D1% = 1 TO dt%          'delay
    NEXT D1%                    'loop

NEXT ii%
ii% = ii% - 1

FOR j% = R1% TO R2%             ' j% is a row variable
LOCATE j%, ii%
PRINT ; "█";
    FOR D1% = 1 TO dt%       'delay
    NEXT D1%                 'loop
NEXT j%

FOR K% = C2% TO C1% STEP -1     'k% is a column variable
  LOCATE j%, K%
  PRINT ; "█";
```

```
    FOR D1% = 1 TO dt%        'delay
    NEXT D1%                  'loop

NEXT K%
K% = K% + 1

FOR 1% = R2% TO R1% STEP -1           '1% is a row variable
  LOCATE 1%, K%
  PRINT ; "█";

    FOR D1% = 1 TO dt%        'delay
    NEXT D1%                  'loop

NEXT 1%
R1% = R1% + 1
C1% = C1% + 1
R2% = R2% - 1
C2% = C2% - 1
IF R1% > 13 GOTO 234

GOTO 123
234
COLOR 7, 0
LOCATE 1, 1
PRINT "Program terminated by user"
END

SUB BLINK (A%(), B%(), MC%, RC%, ROWA%, COLB%, ACBR%)

FOR 1% = 1 TO COLB%                  ' B COLUMNS
BC% = BC1% * 1% + TLCB% - 4          ' CALCULATE LOCATIONS
FOR i% = 1 TO ROWA%                  ' A ROWS
AR% = BR1% * i% + TLRA% - 1          ' FOR ARRAYS
FOR j% = 1 TO ACBR%                  ' A COLUMNS AND B ROWS
AC% = BC1% * j% + TLCA% - 4          '
BR% = BR1% * j% + TLRB% - 1          '''''''

BLNKC% = 14                          ' BLINK COLOR
COLOR 14, 7                         ' EQUATION COLOR
flag% = 1

'*** PROGRAMMER ******************

   ME% = 16     ' ME% moves equation on line 24 left or right

'**********************************

   EC% = j% * ME%
   LOCATE 24, EC%
'''''''
   PRINT USING "(##  x ##)"; A%(i%, j%); B%(j%, 1%);  '
PRINT
```

```
    IF j% < ACBR% THEN                                                 '
                                                                      'EQUATION
     LOCATE 24, EC% + 12                                               ' ON
     PRINT "+";                                                        ' LINE
     ELSE                                                              ' 24
     LOCATE 24, EC% + 12                                               '
     PRINT "= █";                                                      '
     END IF
'''''''''''
    COLOR 15, RC%                                   '
    LOCATE AR%, BC%                                 ' PRINT
    PRINT " █"                                      ' RESULT BOX
COLOR BLNKC% + 16, MC%

10 LOCATE AR%, AC%                            '''''''''''''
    PRINT A%(i%, j%)                          '  BLINK ARRAY ELEMENTS
    LOCATE BR%, BC%                           '  TO BE MULTIPLIED
    PRINT B%(j%, 1%)                          ''''''''
    IF flag% = 0 GOTO 20
DO
LOOP WHILE INKEY$ = ""
    COLOR 15, MC%                               'TURN OFF BLINK
    flag% = 0
    GOTO 10
20
    COLOR 15, RC%                                 ' COLOR OF RESULT BOX
    T% = T% + A%(i%, j%) * B%(j%, 1%)
    NEXT j%
    LOCATE AR%, BC%
    PRINT T%                                          'PRINT TOTAL
    COLOR 15, RC%
    T% = 0
    LOCATE 24, 15                                      ' CLEAR
    COLOR 15, 7                                        '
    PRINT SPC(60);                                     ' LINE 24
    COLOR 15, RC%                                      '
    NEXT i%                                            '
    NEXT 1%                                            '''''''''
END SUB

SUB BLINK2 (A%(), E!(), MC%, RC%, ROWA%, COLB%, ACBR%)

FOR 1% = 1 TO COLB%                             ' B COLUMNS
BC% = BC1% * 1% + TLCB% - 4                     ' CALCULATE LOCATIONS
FOR i% = 1 TO ROWA%                             ' A ROWS
AR% = BR1% * i% + TLRA% - 1                     ' FOR ARRAYS
FOR j% = 1 TO ACBR%                             ' A COLUMNS AND B ROWS
AC% = BC1% * j% + TLCA% - 4                     '
BR% = BR1% * j% + TLRB% - 1                     '''''''

BLNKC% = 14                                     ' BLINK COLOR
COLOR 14, 7                                     ' EQUATION COLOR
```

```
flag% = 1

'*** PROGRAMMER *****************
'
  ME% = 16           ' ME% changes width of equation
'
''*********************************

   EC% = j% * ME%
   LOCATE 24, EC%
'''''''''''
PRINT USING "(##  x ##.##)"; A%(j%, 1%); E!(i%, j%);  '
PRINT
   IF j% < ACBR% THEN                                '
EQUATION
    LOCATE 24, EC% + 14                                ' ON
    PRINT "+";                                         ' LINE
    ELSE                                               ' 24
    LOCATE 24, EC% + 14                                '
    PRINT "=  █";                                      '
    END IF
'''''''''''
   COLOR 15, RC%                        '
   LOCATE AR%, BC%                      ' PRINT
   PRINT " █"                           ' RESULT BOX

COLOR BLNKC% + 16, MC%

910 LOCATE BR%, BC%                ''''''''''''
   PRINT A%(j%, 1%)                '  BLINK ARRAY ELEMENTS
    LOCATE AR%, AC%                '  TO BE MULTIPLIED
   PRINT USING "##.##"; E!(i%, j%)
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

   IF flag% = 0 GOTO 920
DO
LOOP WHILE INKEY$ = ""
   COLOR 15, MC%                   'TURN OFF BLINK
   flag% = 0
   GOTO 910
920
   COLOR 15, RC%                      ' COLOR OF RESULT BOX
   T! = T! + A%(i%, j%) * E!(j%, 1%)

   NEXT j%
   LOCATE AR%, BC%
   PRINT CINT(T!)                              'PRINT TOTAL
   COLOR 15, RC%
   T! = 0
   LOCATE 24, 15                         ' CLEAR
   COLOR 15, 7                           '
   PRINT SPC(60);                        ' LINE 24
   COLOR 15, RC%                         '
   NEXT i%                                      '

   NEXT 1%                                      ''''''''''

END SUB

SUB box (TLR%, TLC%, WIDE%, LL%, FGC%, BGC%, bx$, sfc%,_
style%, syn$)
''''''''''''''''''''''''''''''''''''''''''''''
' TLR%    - Position of Top Left Row
' tlc% - Position of Top Left Column
' wide% - Width of longest line in list
' ll % - number of items in list
' fgc% - Foreground color
' bgc% - Background color
' bx$ - "n" for displaying list
' sfc% - shadow foreground color
' style% - border style
'  syn$ - show shadow? (y)es or (n)o
''''''''''''''''''''''''''''''''''''''''''''''
WIDE% = WIDE% + 2
STLC% = TLC% - 1: STLR% = TLR% + 1: SWIDE% = WIDE% + 4
TALL% = LL% * 2 - 1
'--------------------------------------------------
'   These are the BOX styles available
'--------------------------------------------------
IF style% = 1 THEN
  TL$ = " ╓": TR$ = "╖ ": BL$ = " ╙": BR$ = "╜ ": LS$ = "
║": RS$ = "║ "
LIN$ = STRING$(WIDE%, "─")
ELSEIF style% = 2 THEN
  TL$ = " ┌": TR$ = "┐ ": BL$ = " └": BR$ = "┘ ": LS$ = "
│": RS$ = "│ "
LIN$ = STRING$(WIDE%, " ")
ELSEIF style% = 3 THEN
  TL$ = "  ": TR$ = "  ": BL$ = "  ": BR$ = "  ": LS$ = "
|". RS$ = "| "
LIN$ = STRING$(WIDE%, " ")
END IF
'------------------------------------------------------
TOPLINE$ = TL$ + LIN$ + TR$
BOTLINE$ = BL$ + LIN$ + BR$

IF syn$ = "y" OR syn$ = "Y" THEN

COLOR sfc%, sfc%                                ''''''''''''''
FOR i% = STLR% TO STLR% + TALL% + 1             '
LOCATE i%, STLC%                                '   shadow
PRINT ; SPC(1);                                 '
NEXT i%                                         ''''''''''''''
PRINT ; SPC(SWIDE% - 1);                          '
```

```
END IF

COLOR FGC%, BGC%                                    '''''''''''''''''''
LOCATE TLR%, TLC%                                   '
PRINT ; TOPLINE$;                                   '
FOR i% = TLR% + 1 TO TLR% + TALL%                   '
LOCATE i%, TLC%                                     '      box
PRINT ; LS$; SPC(WIDE%); RS$;                       '
NEXT i%                                             '
LOCATE i%, TLC%                                     '
PRINT ; BOTLINE$;                                   '''''''''''''''''''
IF bx$ = "y" OR bx$ = "Y" GOTO 9999
j% = TLR% + 1
K% = TLC% + 3
FOR i% = 1 TO LL%
LOCATE j%, K%
PRINT ; item$(i%);
j% = j% + 2
NEXT i%
9999
END SUB

SUB clr1 (R1%, C1%, R2%, C2%)

COLOR 7, 7                                   '''''''''''''''''''
FOR n% = 1 TO 6                              '  blanks part of
LOCATE R1% + n% - 2, C1%                     ' screen in the
PRINT SPC(80 - C1%);                         ' color grey
NEXT n%                                      '''''''''''''''''''''

END SUB

SUB clr2 (R1%, C1%, R2%, C2%)

COLOR 7, 7                                   '''''''''''''''''''
FOR n% = R1% TO R2%                          '  blanks part of
LOCATE n%, C1%                              ' screen in the
PRINT SPC(C2% - C1%);                             ' color grey
NEXT n%                                      '''''''''''''''''''''


END SUB

 SUB determ2 (i%, j%, A%(), name$, sol%)

IF name$ = "ROW" THEN
   C% = j% * 24 + 10
   R% = 12
  ELSEIF name$ = "COLUMN" THEN
   C% = i% * 24 + 10
   R% = 12
  ELSEIF name$ = "cofactor" THEN
```

```
   C% = 30
   R% = 16
END IF

COLOR 15, 7
LOCATE R% + 1, C% - 3
PRINT "x"

IF name$ <> "cofactor" THEN
CALL clr1(R%, C% + 11, 1, 1)
END IF

COLOR 15, 7
IF i% <= 2 AND j% <= 2 THEN
LOCATE R% + 1, C% + 7
PRINT "+"
END IF

IF name$ = "ROW" AND j% = 2 OR name$ = "COLUMN" AND i% = 2
THEN
 LOCATE R% + 1, C% + 7
 PRINT "="
 LOCATE R% + 1, C% + 12
 ans% = A%(1, 1) * A%(2, 2) - A%(1, 2) * A%(2, 1)     '
answer
 COLOR 31, 0
 PRINT ans%

END IF

COLOR 0, 7
sign% = (-1) ^ (i% + j%)
IF sign% = 1 THEN sign$ = "( 1)"
IF sign% = -1 THEN sign$ = "(-1)"

LOCATE R% + 1, C% - 12              'print sign next to minor
PRINT sign$

COLOR 14, 4
LOCATE R% + 1, C% - 8              'print intersection next to
minor
PRINT A%(i%, j%)

COLOR 15, 1
'-------------------
IF i% = 1 AND j% = 1 THEN
'-------------------

LOCATE R% + 1, C%
PRINT A%(i% + 1, j% + 1)
END IF
'-------------------
```

```
IF i% = 1 AND j% = 2 THEN
'-------------------
LOCATE R% + 1, C%
PRINT A%(i% + 1, j% - 1)
END IF
'-------------------
IF i% = 2 AND j% = 1 THEN
'-------------------

LOCATE R% + 1, C%
PRINT A%(i% - 1, j% + 1)
END IF
'-------------------
IF i% = 2 AND j% = 2 THEN
'-------------------
LOCATE R% + 1, C%
PRINT A%(i% - 1, j% - 1)
END IF
END SUB

SUB iandj (i%, j%, hfc%, hbc%, ijr%, ijc%, ijfc%, ijbc%)
    COLOR ijfc%, ijbc%                       ''''''''''''''''''''
    LOCATE ijr% + 1, ijc% + 4                '  i and j values
    PRINT "i = "; i%                         '
    LOCATE ijr% + 3, ijc% + 4                '  i and j values
    PRINT "j = "; j%                         '
    COLOR hfc%, hbc%                         ''''''''''''''''''''

END SUB

SUB minor (i%, j%, A%(), name$, sol%, tsol%)


IF name$ = "ROW" THEN
 C% = j% * 27 - 12
ELSEIF name$ = "COLUMN" THEN
 C% = i% * 27 - 12
ELSEIF name$ = "cofactor" THEN
 C% = 37
END IF
 R% = 16

CALL box(R% - 1, C% - 2, 7, 2, 15, 1, "y", 0, 3, "y")
'print minor box

COLOR 15, 7
LOCATE R% + 1, C% - 5
PRINT "x"

IF name$ <> "cofactor" THEN
 CALL clr2(7, C% + 11, 20, 80)   'clear all minors and
results if i or j = 1
```

```
END IF

COLOR 15, 7

IF i% < 3 AND j% < 3 THEN
LOCATE R% + 1, C% + 12              'print + or = sign next to
minor

 IF name$ = "cofactor" THEN
     PRINT " ="
    ELSE
     PRINT "+"
 END IF

END IF
'----------------------------------------------------------
'The next 2 IF statements ensure correct placement
'of "+" next to minor boxes
IF name$ = "ROW" AND i% = 3 AND j% < 3 THEN
'
LOCATE R% + 1, C% + 12       'print + sign next to minor     '
PRINT "+"
END IF
IF name$ = "COLUMN" AND j% = 3 AND i% < 3 THEN
LOCATE R% + 1, C% + 12        'print + sign next to minor
PRINT "+"
END IF
'---------------------------------------------------------

 COLOR 0, 7
 sign% = (-1) ^ (i% + j%)
 IF sign% = 1 THEN sign$ = "( 1)"
 IF sign% = -1 THEN sign$ = "(-1)"

 LOCATE R% + 1, C% - 13          'print sign next to minor
 PRINT sign$

IF name$ <> "cofactor" THEN
COLOR 14, 4
LOCATE R% + 1, C% - 9         'print intersection next to
minor
A%(i%, j%) = A%(i%, j%)
PRINT A%(i%, j%)
END IF

COLOR 15, 1

'------------------
IF i% = 1 AND j% = 1 THEN
'------------------
d11% = A%(i% + 1, j% + 1)
d22% = A%(i% + 2, j% + 2)
```

```
d21% = A%(i% + 2, j% + 1)
d12% = A%(i% + 1, j% + 2)

'-------------------
ELSEIF i% = 1 AND j% = 2 THEN
'-------------------
d11% = A%(i% + 1, j% - 1)
d22% = A%(i% + 2, j% + 1)
d21% = A%(i% + 2, j% - 1)
d12% = A%(i% + 1, j% + 1)

'-------------------
ELSEIF i% = 1 AND j% = 3 THEN
'-------------------
d11% = A%(i% + 1, j% - 2)
d12% = A%(i% + 1, j% - 1)
d21% = A%(i% + 2, j% - 2)
d22% = A%(i% + 2, j% - 1)

'-------------------
ELSEIF i% = 2 AND j% = 1 THEN
'-------------------
d11% = A%(i% - 1, j% + 1)
d12% = A%(i% - 1, j% + 2)
d21% = A%(i% + 1, j% + 1)
d22% = A%(i% + 1, j% + 2)

'-------------------
ELSEIF i% = 2 AND j% = 2 THEN
'-------------------
d11% = A%(i% - 1, j% - 1)
d12% = A%(i% - 1, j% + 1)
d21% = A%(i% + 1, j% - 1)
d22% = A%(i% + 1, j% + 1)

'-------------------
ELSEIF i% = 2 AND j% = 3 THEN
'-------------------
d11% = A%(i% - 1, j% - 2)
d12% = A%(i% - 1, j% - 1)
d21% = A%(i% + 1, j% - 2)
d22% = A%(i% + 1, j% - 1)

'-------------------
ELSEIF i% = 3 AND j% = 1 THEN
'-------------------
d11% = A%(i% - 2, j% + 1)
d12% = A%(i% - 2, j% + 2)
d21% = A%(i% - 1, j% + 1)
d22% = A%(i% - 1, j% + 2)

'-------------------
```

```
ELSEIF i% = 3 AND j% = 2 THEN
'------------------
d11% = A%(i% - 2, j% - 1)
d12% = A%(i% - 2, j% + 1)
d21% = A%(i% - 1, j% - 1)
d22% = A%(i% - 1, j% + 1)

'------------------
ELSEIF i% = 3 AND j% = 3 THEN
'------------------
d11% = A%(i% - 2, j% - 2)
d12% = A%(i% - 2, j% - 1)
d21% = A%(i% - 1, j% - 2)
d22% = A%(i% - 1, j% - 1)

END IF

LOCATE R%, C%
PRINT d11%
LOCATE R%, C% + 6
PRINT d12%
LOCATE R% + 2, C%
PRINT d21%
LOCATE R% + 2, C% + 6
PRINT d22%
                                        '''''''''''''''''
 ans% = d11% * d22% - d21% * d12%       '1. determinate of
                                        '   cofactor
 bns% = (-1) ^ (i% + j%) * A%(i%, j%)  '2. cofactor element
 sol% = ans% * bns%                     '3. solution
                                        ''''''''''''''''''''

tsol% = tsol% + sol%

LOCATE R% + 4, C% + 2
''''''''''''''''''''''''''''''''''''
COLOR 7, 0                             'Print solution to 2x2
                                        'determinent
PRINT ans%                             'in the lower shadow
COLOR 15, 1                            '
                                       ''''''''''''''''''''''

'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
IF name$ = "cofactor" THEN
''''''''''''''''''''''''

 ans% = d11% * d22% - d21% * d12%      ' calculate and print
 bns% = (-1) ^ (i% + j%)               ' * A%(i%, j%)
                                        ' cofactor element
 sol% = ans% * bns%                       ' solution

 LOCATE R% + 1, 54
 COLOR 7, 7
 PRINT "           "
                                        '''''''''''''''''
 COLOR 15, 4                                         'color of box
 LOCATE R% + 1, 54                                   '
                                                     ''''''''''''''''''
 PRINT "█"                                           'print solution box
                                                     '''''''''''''''''''
END IF
'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
IF name$ = "ROW" THEN                             '   position the
                                                  'intermediate
 C% = j% * 13 + 8                                 '
ELSEIF name$ = "COLUMN" THEN                      '  solution in relation
to
 C% = i% * 13 + 8                                 '
ELSEIF name$ = "cofactor" THEN GOTO 331       '    the MATRIX
 C% = 37                                          '
END IF                                            '
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
COLOR 15, 7
LOCATE R% - 7, 25
PRINT "="

IF name$ = "ROW" AND j% < 3 THEN

COLOR 0, 7
LOCATE R% - 7, C% + 10
' print intermediate solution
PRINT ; sol%; : COLOR 15, 7: PRINT ; SPC(4); "+"

ELSEIF name$ = "COLUMN" AND i% < 3 THEN

COLOR 0, 7
LOCATE R% - 7, C% + 10
' print intermediate solution
PRINT ; sol%; : COLOR 15, 7: PRINT ; SPC(4); "+"

ELSE

COLOR 0, 7
LOCATE R% - 7, C% + 10
PRINT ; sol%; : COLOR 15, 7: PRINT ; SPC(4); "="; SPC(4);
COLOR 31, 0
PRINT tsol%

IF tsol% = 0 THEN
COLOR 20, 7
LOCATE R% - 5, C% + 22
PRINT "Oh, Oh!"
LOCATE R% - 3, C% + 18
COLOR 0, 7
PRINT "Why is this 0?"
```

```
END IF

tsol% = 0

END IF
'**********************************
331

END SUB
```

<u>Appendix C</u>: <u>VVAM Protocol Handout</u>

SUGGESTIONS FOR COMPLETING THE HANSARD
INTERACTIVE MATRIX ALGEBRA EXERCISE

First and Foremost:

This exercise is DESIGNED TO FACILITATE SHARING OF INSIGHTS between YOU and YOUR INSTRUCTOR.

IT IS NOT DESIGNED TO BE USED as a stand alone Computer Assisted Instruction (CAI) package.

We expect a HUMAN INSTRUCTOR to be at the ready to answer any and all questions YOU, as a STUDENT of Matrix Algebra, may need answered while using the package. We also expect the INSTRUCTOR to help YOU, as a student, to EFFECTIVELY USE the package,

WE'RE SURE MEANINGFUL LEARNING IS INSPIRED BY A CONSTRUCTIVE PARTNERSHIP OF STUDENT AND TEACHER!

If we are right, let us know.

If not, well, we're ready to hear that too!

We certainly hope the EXERCISES offered by the program motivate an INTENSE DIALOGUE about the SUBJECT of MATRIX ALGEBRA BETWEEN YOU AND YOUR TEACHER and give YOU personally, a chance to CONCRETELY EXPERIENCE what 'working with matrices' really involves.

OUR MOTTO IS: It's the PROCESS,
NOT the PRODUCT!

We suggest, for each MATRIX ACTIVITY (addition, subtraction, multiplication, inversion, etc...), that you try to follow the PROTOCOL we list below. Please treat our PROTOCOL as a HEURISTIC, and not as an ALGORITHM. Vary DETAILS OF EACH STEP IN THE PROTOCOL, as YOUR intuition and experience suggest.

PLEASE feel free to SHARE what you learn and discover with us. Believe me, we are very VERY INTERESTED.

GOOD LUCK!

A SUGGESTED PROTOCOL FOR PARTICIPANTS IN THE INTERACTIVE LEARNING OF APPLIED LINEAR ALGEBRA

Caution: Don't rush. Take your time. Complete every suggested step of this protocol and OBJECTIFY EVERYTHING... on paper, on a blackboard. Don't assume anything without RECORDING THAT FACT on a medium others can SEE! If you 'DO KNOW IT' then WRITE IT DOWN so others can CONFIRM you do... EXPLICITLY and COMPLETELY!

1) Exercise the PROGRAM for a SPECIFIC TECHNIQUE, starting with EXERCISE 1 - Matrix Addition.

ASK THE PROGRAM TO PERFORM THE MATRIX OPERATION REQUESTED! Watch it operate, CAREFULLY! Lots of colorful images have been created to help you FOLLOW THE ACTION. Do just that - carefully.

WHY? Because, as of Step 2, we will be asking you to VERBALIZE WHAT YOU HAVE SEEN - every single thing. So OBSERVE and RECORD what you SEE mentally, with GREAT CARE!

Carry out THE SPECIFIC MATRIX OPERATION and vary the size of the matrices. Practice all you want to. RUN the program for a particular operation as many times as you wish. OBSERVE, RECORD, FOLLOW EVERY STEP... EVERY SINGLE STEP. Take note of the numbers and colors and items before your eyes. Everything is significant - EVERYTHING!

Don't be discouraged, or frustrated, IF YOU HAVE TO START AGAIN. Go right ahead... as many times as you please.

There is absolutely no penalty for starting over. As a matter of fact, it has been our experience with this package, that STUDENTS LEARN A GREAT DEAL IF THEY ARE WILLING TO TAKE RISKS, and MAKE MISTAKES. Just try not make the SAME MISTAKE twice. If you do... so, BIG DEAL. That happens. Enjoy.

2) VERBALIZE, in STRAIGHT FORWARD ENGLISH, EXACTLY WHAT YOU OBSERVE HAPPENING.

Ask YOUR TEACHER TO LISTEN to your description with HIS/HER EYES CLOSED (and no cheating!!!)

Be sure YOUR TEACHER
can follow every single step,
and we mean -- EVERY SINGLE STEP!

Work HARD at DESCRIBING THE OPERATION, with PRECISION and without leaving out anything! VERBALIZE every move and continue to do so until you ARE ABSOLUTELY SURE YOUR TEACHER understands, with EYES CLOSED, EVERY SINGLE THING YOU HAVE SAID - in PRECISELY THE ORDER YOU SAID IT. We caution you: Take nothing for granted! And DON'T HURRY. Take the TIME to DO IT RIGHT - no matter how much time that may take. Make time work for you. That takes PRACTICE! Sometimes a little, sometimes a lot. Take whatever time necessary!

3) Using a blackboard or piece of paper, develop a RIGOROUS ALGORITHM for what you have VERBALIZED in Step 2. Use your own PSEUDO-CODE [a programming like language that allows you to specify INPUT/OUTPUT, a SEQUENCE of instructions, BRANCHING (UNCONDITIONAL and CONDITIONAL) and ITERATION (LOOPING through a given set of instructions more than once)].

WRITE DOWN - I SAID WRITE DOWN!!!
ON THE BLACKBOARD OR PIECE OF PAPER, WHERE IT CAN BE SEEN BY OTHERS!!!!!
a COMPLETE ALGORITHM for the OPERATION UNDER STUDY.

Make sure - BY ACTIVE CHECKING AND EXECUTION OF YOUR ALGORITHM - THAT IT WORKS - EVERY TIME - FOR ALL SITUATIONS covered by the HANSARD SYSTEM - AT LEAST. Ideally, it should work for ANY SET OF OF MATRICES - GIVEN whatever ASSUMPTIONS you have specified in your ALGORITHMIC SPECIFICATION.

USE CONCEPT MAPS and the V-HEURISTIC to your advantage when constructing your algorithm. Be sure to clearly AND COMPLETELY specify:

THE ALGORITHM and ALL DATA STRUCTURES required to carry out the INTENDED OPERATION.

Use the images produced by the HANSARD system to stimulate ideas about the algorithm you need and the data structures with which the algorithm will work.

4) Once you

1 - are able to verbalize what operations are required to perform the matrix operation at hand,

and, 2 - you have a rigorous and BUG FREE algorithm to direct a 3rd party to perform the matrix operation GIVEN some specific matrices,

THEN:

Develop a MATHEMATICALLY RIGOROUS MODEL (with appropriate formulae) that specify MATHEMATICALLY how to perform the OPERATION under study.

This will typically involve using the SUMMATION OPERATOR of ALGEBRA (SIGMA) and a set of VARIABLES - including the INDEX of SUMMATION.

If you know MathCAD... you are welcome to construct this model in MathCADic FORM. Otherwise, specify the Mathematical formulae in ALGEBRAIC FORM.

Try your Mathematical Model out.

Make sure - IT WORKS IN THEORY AND PRACTICE!

When you have done ALL that... you can give yourself a big pat on the back.

You are THINKING LIKE A MATHEMATICIAN!

We ESTIMATE a total of 6-8 hours will be required to carry out this PROTOCOL on ALL EXERCISES OFFERED BY the HANSARD SYSTEM. We also expect WIDE VARIATION around this MEAN ESTIMATE since the BACKGROUND and LEARNING SKILLS of students and teachers will vary widely. Both Stone and I will be delighted to hear your SUGGESTIONS FOR IMPROVEMENT

of the Displays,
of the Program Logic,
of the Protocol,
of ANYTHING ELSE you can think of!!!

Mail your suggestions to:

DREYNOLDS or SHANSARD on the AFIT CSC VAX.

And thank YOU... for helping us carry out research we hope will produce PEDAGOGY RIGHT FOR THE TIMES and RIGHT FOR YOU!

Now... get to work!!!

Appendix D: Thesis Questionnaire Source Code

```
DECLARE SUB box (TLR%, TLC%, WIDE%, LL%, FGC%, BGC%, bx$,
sfc%, style%, syn$) DECLARE SUB getinput (instr$, maxlen%)
DIM textline$(40, 3)               ' (n,m) n=question
number, m=line number DIM SHARED item$(5)
DIM SAVECAR%(10100), WALL%(900), MAN%(500), HEAD%(500),
STONE%(10000), PRO%(500)

CLEAR , , 10000                                'Increase stack
size '''''''''''''''''''''''''''''''''''''''''''''''
' TLR%    - Position of Top Left Row
' tlc% - Position of Top Left Column
' wide% - Width of longest line in list
' ll % - number of items in list
' fgc% - Foreground color
' bgc% - Background color
' bx$ - "n" for displaying list
' sfc% - shadow foreground color
' style% - border style
' syn$ - show shadow? (y)es or (n)o
'''''''''''''''''''''''''''''''''''''''''''''''
'DIM SHARED item$(12)
ans1$ = ""          ''''''''''''''''''''''''''
ans2$ = ""          '  Make sure all answers
ans3$ = ""          '  are empty before
ans4$ = ""          '  inputs are made
ans5$ = ""          ''''''''''''''''''''''''''
COLOR 15, 7
CLS
'----------------------------------------------------------
item$(1) = "T H E S I S   Q U E S T I O N A I R E"
CALL box(2, 20, 37, 1, 15, 2, "n", 0, 1, "y")
'----------------------------------------------------------
'----------------------------------------------------------
CALL box(7, 5, 68, 8, 11, 1, "y", 0, 1, "y")
'----------------------------------------------------------
COLOR 15
r% = 10
c% = 8
LOCATE r% - 1, c%
PRINT "      THANK YOU FOR PARTICIPATING IN THIS LEARNING
EXERCISE!" LOCATE r%, c%
PRINT "
___________________________________________________________"

LOCATE r% + 3, c%

PRINT "        Please answer the following two sets of
questions;    " LOCATE r% + 5, c%
```

```
PRINT "your responses will help us understand the value of
this experiment." LOCATE r% + 7, c%

PRINT "     The first set of questions is a short
demographic survey." LOCATE r% + 9, c%

PRINT "    The second set of questions lets you tell us what
you think" LOCATE r% + 11, c%

PRINT "                    of this LEARNING/TEACHING
EXPERIENCE."

LOCATE 24, c%
COLOR 14, 0
PRINT ; "                         <HIT ANY KEY TO BEGIN>";

DO
LOOP UNTIL INKEY$ <> ""
COLOR 15, 7
CLS
'-----------------------------------------------------------
item$(1) = "     D E M O G R A P H I C S "
CALL box(2, 22, 32, 1, 15, 4, "n", 0, 1, "y")
'-----------------------------------------------------------
COLOR 15, 7
'-------------------------
CALL box(8, 7, 63, 5, 15, 1, "Y", 0, 1, "y")
'---------------------------------------------
r% = 11
LOCATE r%, 10
PRINT " Your answers to this set of questions will provide
background" LOCATE r% + 1, 10
PRINT "      information to help us correlate your responses
to" LOCATE r% + 2, 10
PRINT "                  the second set of questions."
LOCATE r% + 4, 10
PRINT "If you don't want to answer a question, hit <Enter>
to skip it."



LOCATE r% + 8, c%
COLOR 14, 0
PRINT ; "                         <HIT ANY KEY TO CONTINUE>";
DO
LOOP UNTIL INKEY$ <> ""
CLS
'-----------------------------------------------------------
item$(1) = "     D E M O G R A P H I C S "
CALL box(1, 22, 32, 1, 15, 1, "n", 0, 1, "y")
'-----------------------------------------------------------
```

```
2
row% = 6: colm% = 6: LIN% = 5
COLOR 7, 0

LOCATE row%, colm%
PRINT ; "1.  What is your AFIT program (GSM, GLM, etc)?  ";
CALL getinput(ans1$, LIN%)
COLOR 7

LOCATE row% + 2, colm%
PRINT ; "2.  What is your level of education (BS, MA, etc)?
"; CALL getinput(ans2$, 3)
COLOR 7

LOCATE row% + 4, colm%
PRINT ; "3.  What was the subject of your latest degree?  ";
CALL getinput(ans3$, 22)
COLOR 7

LOCATE row% + 6, colm%
PRINT ; "4.  Have you taken a course in MATRIX MATH (Y/N)?
"; CALL getinput(ans4$, 2)
COLOR 7

LOCATE row% + 8, colm%
PRINT ; "5.  Have you learned MATRIX MATH in conjunction
with" LOCATE row% + 9, colm%
PRINT ; "    any course (Y/N)?  ";
CALL getinput(ans5$, 2)
COLOR 7

LOCATE row% + 11, colm%
PRINT ; "6.  Please enter your MBTI code.  ";
CALL getinput(ans6$, 5)
COLOR 7

'---------------------------------------
CALL box(20, 4, 68, 2, 15, 4, "y", 0, 1, "n")
'---------------------------------------
c% = 0
r% = 21
LOCATE r%, c% + 7
PRINT "Please review your answers and make one of the
following selections.";

c% = 4
LOCATE r% + 2, c% + 7
PRINT "       (C)hange an answer   -   (B)egin the
questionaire " COLOR 14
LOCATE r% + 2, c% + 14

PRINT "(C)"
```

```
LOCATE r% + 2, c% + 37
PRINT "(B)"


391
DO
sel$ = INKEY$
LOOP UNTIL sel$ <> ""
IF sel$ = "b" OR sel$ = "B" GOTO 667

IF sel$ = "c" OR sel$ = "C" GOTO 392

GOTO 391

392
'-----------------------------------------
CALL box(20, 4, 68, 2, 15, 4, "y", 0, 1, "y")
'-----------------------------------------
LOCATE r%, c% + 3
PRINT "If an answer is correct, hit <Enter> to move to the
next question. " LOCATE r%, c% + 32
COLOR 14, 4
PRINT "<Enter>"
COLOR 15, 4
LOCATE r% + 2, c% + 3
PRINT "To change an answer, just retype it.  "

GOTO 2
667
OPEN "tq.out" FOR APPEND AS #6
PRINT #6, ans1$
PRINT #6, ans2$
PRINT #6, ans3$
PRINT #6, ans4$
PRINT #6, ans5$
PRINT #6, ans6$

COLOR 7, 7
CLS
OPEN "tquest .inp" FOR INPUT AS #5
'----------------------------------------------
item$(1) = "   Q U E S T I O N A I R E   "
 CALL box(1, 25, 29, 1, 15, 2, "n", 0, 1, "y")
'----------------------------------------------
 qn% = 1
 TR% = 24
 tc% = 6

 COLOR 0, 0
 LOCATE 16, 1
 FOR j% = 1 TO 8
 PRINT ; "
";
 NEXT j%

 COLOR 9, 0
 LOCATE 16, 1
 PRINT ;
"╔══════════════════════════════════════════════════════════
═════════════════════╗";
 FOR 1% = 17 TO 23
 LOCATE 1%, 1
 PRINT ; "║
║";
 NEXT 1%
 LOCATE 24, 1
 PRINT ;
"╚══════════════════════════════════════════════════════════
═════════════════════╝";
 COLOR 11, 0
 LOCATE 25, 1
 PRINT ; "          Use <Arrow> keys to select response -
<Enter> to enter response        ";
 LOCATE 25, 14
 COLOR 14
 PRINT ; "<Arrow>";
 LOCATE 25, 48
 PRINT ; "<Enter>";
 COLOR 9, 0
'-------------------------------------------------------------
------------
 r% = 6
 c% = 10
'----------------------------------------------
 CALL box(r%, c%, 57, 3, 15, 4, "y", 0, 1, "y")'question box
'----------------------------------------------

1
 FOR i% = 1 TO 3
 IF EOF(5) THEN
CLOSE #5
CLOSE #6
   r1% = 1
   c1% = 1
   r2% = 24
   c2% = 80
   klr% = 0
LOCATE r1%, c1%

123
  COLOR klr%
  FOR ii% = c1% TO c2%        ' i% is a column variable
  LOCATE r1%, ii%
  PRINT ; "█";

    FOR d% = 1 TO 100     'delay
    NEXT d%               'delay

NEXT ii%
ii% = ii% - 1

FOR j% = r1% TO r2%              ' j% is a row variable
LOCATE j%, ii%
PRINT ; "█";
    FOR d% = 1 TO 100     'delay
    NEXT d%               'delay
NEXT j%

FOR K% = c2% TO c1% STEP -1     'k% is a column variable
  LOCATE j%, K%
  PRINT ; "█";

    FOR d% = 1 TO 100     'delay
    NEXT d%               'delay

NEXT K%
K% = K% + 1

FOR l% = r2% TO r1% STEP -1                    'l% is a row
variable
  LOCATE l%, K%
  PRINT ; "█";

    FOR d% = 1 TO 100     'delay
    NEXT d%               'delay

NEXT l%
r1% = r1% + 1
c1% = c1% + 1
r2% = r2% - 1
c2% = c2% - 1
IF r1% > 13 GOTO 234

GOTO 123
234

SCREEN 9, , 0, 1
COLR% = 8
pi# = 3.1416
COLOR 12, COLR%
LOCATE 2, 32
PRINT "THIS HAS BEEN A"

COLOR 12, COLR%
FOR K% = 0 TO 10 STEP 5
```

```
IF K% = 0 THEN co% = 11
IF K% = 5 THEN co% = 12
IF K% = 10 THEN co% = 10
CIRCLE (50, 250), 20, co%, 0, 3 * pi# / 2    '  S
CIRCLE (50, 280), 20, co%, pi#, pi# / 2        '  S
DRAW "BU45 BR25 R50 L25 D60"               't
CIRCLE (150, 265), 29, co%, , , 1.5  '        'O
DRAW "BD30 BR40 U60 TA25 D65 TA0 U60"       'N
DRAW "BR25 R35 L35 D30 R25 L25 D30 R35"     'E
DRAW "BU60 BR20 D60 TA155 D40 TA25 D40 TA0 U60 "    'W
DRAW "BD60 BR20 TA-15 U62 TA15 D62 U25 TA0 L25"     'A
DRAW "BU36 BR50 D60 R35 BR15 BU60 D60 R35 C15"      'L L
GET (20, 230)-(520, 300), STONE%
PUT (20, 230), STONE%

FOR j% = 1 TO 5 STEP 1
PUT (60 + K% + j%, 60 + K% + j%), STONE%, XOR

PCOPY 0, 1
NEXT j%
NEXT K%
PX% = POINT(0)
PY% = POINT(1)

'SCREEN 9, , 0, 1
DRAW "BM320,175"
COLOR 1, COLR%
pi# = 3.1416
PX% = POINT(0)
PY% = POINT(1)

GOSUB WALL
DRAW "BM320,175"
GOSUB CAR
DRAW "BM320,175"

GOSUB MAN
PCOPY 0, 1
FOR j% = 1 TO 63 STEP 5
COLOR 15, j%
PCOPY 0, 1
FOR 1% = 1 TO 2
NEXT 1%
NEXT j%
COLOR 15, 8
LOCATE 14, 32
PRINT "P R O D U C T I O N"
GET (245, 183)-(410, 193), PRO%
PUT (245, 183), PRO%, XOR
FOR H% = 1 TO 220 STEP 8
PUT (450 - H%, 183), PRO%, PSET
PUT (450 - H%, 183), PRO%, OR
```

```
PCOPY 0, 1
NEXT H%
LOCATE 24, 29
COLOR 14
PRINT ; "< Hit any key to Quit >";
PCOPY 0, 1
DO
LOOP UNTIL INKEY$ <> ""

GOTO 666
'END
'---------
MAN:
'---------

CIRCLE (320, 220), 4, 4          '?
PAINT (320, 220), 2, 4           '?
B$ = " S4 TA20 L20 TA0 L15 R15 TA45 L15 R15 TA20 R10 TA0 L10
R10 TA60 L10" DRAW B$

PUT (X%, Y%), SAVECAR%, XOR
PUT (320, 235), WALL%, PSET
'LINE (315, 215)-(330, 225), 15, B
 GET (315, 215)-(330, 224), HEAD%
 GET (280, 215)-(330, 234), MAN%
PCOPY 0, 1
PUT (280, 215), MAN%, XOR

   GOSUB wndw

   MY% = 200
 FOR MX% = 280 TO 375 STEP 10

   IF MX% > 310 THEN
                                             ''''''''''''
    MY% = MY% + 9                            ' MAN FALLS
    END IF                                   ',,,,,,,,,,,

      PUT (MX%, MY%), MAN%
      PCOPY 0, 1
      PUT (MX%, MY%), MAN%, XOR
 NEXT MX%
BODY$ = "S6 BD20 C12 TA45 R5 TA-25 R5 TA35 R5 TA-55 R5 TA65
R5 TA-35 R5 TA0" BODY1$ = " C10 BL+25 TA20 R5 TA-45 R5 TA45
R5 TA-20 R5 TA45 R5 TA-45 R5 S4" DRAW BODY$ + BODY1$
      PCOPY 0, 1
'-----------------------------------------------
PUT STEP(1, -6), HEAD%                     'HEAD APPEARS
      PCOPY 0, 1
PUT STEP(0, 0), HEAD%, XOR
      PCOPY 0, 1
```

```
FOR HX% = 450 TO 620 STEP 10
PUT (HX%, 260), HEAD%, XOR
PCOPY 0, 1
PUT (HX%, 260), HEAD%, XOR
NEXT HX%
PUT (620, 260), HEAD%, OR

RETURN
END
'-------
wndw:
'----------
DRAW "BM285,235"
W$ = " TA-18 R10 TA54 L5 "
DRAW "S2 C15" + W$
DRAW "S3" + W$
DRAW "S4 C12" + W$
'DRAW "BM320,175"
RETURN
END

'--------
CRASH:
'---------

DRAW "BM 310,260"
STAR$ = " C15 TA-18 U5 TA18 D5 TA54 U5 TA-90 U5 TA-54 D5
TA0" DRAW "S2" + STAR$
DRAW "S10" + STAR$
DRAW "S15" + STAR$
DRAW "S20" + STAR$
COLOR 1, 15
FOR 1% = 1 TO 5000
NEXT 1%
COLOR 1, 12
FOR 1% = 1 TO 5000
NEXT 1%

RETURN
END

'--------------
CAR:
'---------------

DRAW "BM10,235"
CAR1$ = "C15 R20 TA45 R20 TA0 R30 TA-45 R20 TA-10 R25 TA-60
R15 TA0 L115 TA75 R18" CAR2$ = "C15 TA0 R76 BL10 BU2 P8,15
BD2 BR10"
DRAW CAR1$ + CAR2$
PAINT STEP(-10, 2), 4, 15
```

```
CIRCLE STEP(25, 11), 10, 15, -pi#, 0                    'FRONT
WHEEL PAINT STEP(2, 2), 8, 15
CIRCLE STEP(-74, -2), 10, 15, -pi#, 0                  ' REAR
WHEEL PAINT STEP(2, 2), 8, 15
'LINE (310, 160)-(432, 196), 2, B                  ' BOX THE CAR

LY% = 235

GET (1, LY% - 10)-(132, LY% + 36), SAVECAR%
PUT (1, LY% - 10), SAVECAR%, XOR

Y% = LY%

FOR X% = 1 TO 210 STEP 10          ' STEP 10 NORMAL-  STEP 30
FOR COLORS PUT (X%, Y%), SAVECAR%, XOR

IF X% > 210 THEN
PUT (320, 235), WALL%, PSET
GOSUB CRASH
COLOR 1, 8
END IF

PCOPY 0, 1
PUT (X%, Y%), SAVECAR%, XOR

NEXT X%
PUT (320, 235), WALL%, PSET
GOSUB CRASH
COLOR 1, 8
RETURN
END
'---------------
WALL:
'---------------

WALL$ = " U5 R30 D10 L30 U5 BR1 P4,15 BL1 C15 R30 L15 U5 D5
L7 D5 R15 U5"

DRAW WALL$
GET (320, 170)-(380, 180), WALL%
PUT (320, 170), WALL%, XOR
FOR j% = 235 TO 255 STEP 10
PUT (320, j%), WALL%, OR
NEXT j%
GET (320, 235)-(370, 265), WALL%
RETURN
END
 END IF

 LINE INPUT #5, textline$(1, i%)

 NEXT i%
 LINE INPUT #5, dum$

 r% = 6
 c% = 10

 COLOR 4, 4

 FOR i% = 3 TO 5
  LOCATE r% + i%, c% + 3
  PRINT ; "
";
 NEXT i%

 COLOR 14, 4
 LOCATE r% + 1, c% + 3
 PRINT "STATEMENT #"; qn%; ":"
 COLOR 15, 4
 FOR i% = 1 TO 3
 LOCATE r% + i% + 2, c% + 3
 PRINT textline$(1, i%)
 NEXT i%
  chooz% = 3
''''''''''''''''''''''''''''''''''''''''''''''''''''
 r% = 18      ' location of
 c% = 5       ' Likert scale

''''''''''''''''''''''''''''''''''''''''''''''''''''''
GOSUB likert
GOSUB choozit

24        'return here

KEY 17, CHR$(&H0) + CHR$(28)         'enter - ?
KEY 15, CHR$(&H0) + CHR$(1)         'esc - QUIT
       KEY(12) ON
       KEY(13) ON
       KEY(15) ON
       KEY(17) ON
29
    ON KEY(12) GOSUB MOVELT
    ON KEY(13) GOSUB MOVERT
    ON KEY(15) GOSUB ext
    ON KEY(17) GOSUB rtn

GOTO 29

choozit:
 IF chooz% = 1 THEN
 COLOR 28, 0                         ' 28 = blink red
 LOCATE r% + 4, c% + 3
 PRINT "^"
 LOCATE r% + 1, c%
 PRINT "Strongly"
 LOCATE r% + 2, c%
 PRINT "Disagree"
ELSEIF chooz% = 2 THEN
 COLOR 28, 0                                           ' 28 = blink
red
 LOCATE r% + 4, c% + 19
 PRINT "^"
 LOCATE r% + 2, c% + 16
 PRINT "Disagree"
ELSEIF chooz% = 3 THEN
 COLOR 27, 0                                           ' 27 = blink
bright cyan
 LOCATE r% + 4, c% + 35
 PRINT "^"
 LOCATE r% + 2, c% + 31
 PRINT "Undecided"
ELSEIF chooz% = 4 THEN
 COLOR 26, 0                                           ' 26 = blink
green
 LOCATE r% + 4, c% + 51
 PRINT "^"
 LOCATE r% + 2, c% + 49
 PRINT "Agree"
ELSEIF chooz% = 5 THEN
 COLOR 26, 0                                           ' 26 = blink
green
 LOCATE r% + 4, c% + 67
 PRINT "^"
 LOCATE r% + 1, c% + 63
 PRINT "Strongly"
 LOCATE r% + 2, c% + 65
 PRINT "Agree"

 END IF
 COLOR 15
 RETURN
 END
'=========================================================
MOVELT:                                    ' move  left
'=========================================================
       KEY(12) OFF
       KEY(13) OFF
       KEY(15) OFF
       KEY(17) OFF
 GOSUB likert
 chooz% = chooz% - 1
 IF chooz% < 1 THEN chooz% = 1
 GOSUB choozit
 GOTO 24
 RETURN
 END
```

```
'==========================================================
MOVERT:                                  ' move  right
'==========================================================
       KEY(12) OFF
       KEY(13) OFF
       KEY(15) OFF
       KEY(17) OFF
 GOSUB likert
 chooz% = chooz% + 1
 IF chooz% > 5 THEN chooz% = 5
 GOSUB choozit
 GOTO 24
 RETURN
 END
 '==========================================================
likert:                                  ' white scale
'==========================================================

 COLOR 15, 0
 LOCATE r% + 1, c%
 PRINT "Strongly
Strongly"
 LOCATE r% + 2, c%
 PRINT "Disagree        Disagree      Undecided
Agree            Agree"
 LOCATE r% + 3, c%
 PRINT "
├───────────────┼───────────────┼───────────────┼──────────
──────┤"
 LOCATE r% + 4, c%
 PRINT "                                                          "
 RETURN
 END
ext:
       KEY(12) OFF
       KEY(13) OFF
       KEY(15) OFF
       KEY(17) OFF
GOTO 666
RETURN
END
rtn:
       KEY(12) OFF
       KEY(13) OFF
       KEY(15) OFF
       KEY(17) OFF
       PRINT #6, chooz%
       qn% = qn% + 1
GOTO 1
RETURN
END
```

```
666
CLS
SCREEN 0
END

SUB box (TLR%, TLC%, WIDE%, LL%, FGC%, BGC%, bx$, sfc%,
style%, syn$)
''''''''''''''''''''''''''''''''''''''''''''''
' TLR%    - Position of Top Left Row
' tlc% - Position of Top Left Column
' wide% - Width of longest line in list
' ll % - number of items in list
' fgc% - Foreground color
' bgc% - Background color
' bx$ - "n" for displaying list
' sfc% - shadow foreground color
' style% - border style
'  syn$ - show shadow? (y)es or (n)o
''''''''''''''''''''''''''''''''''''''''''''''
WIDE% = WIDE% + 2
STLC% = TLC% - 1: STLR% = TLR% + 1: SWIDE% = WIDE% + 4
TALL% = LL% * 2 - 1
'------------------------------------------------
'   These are the BOX styles available
'------------------------------------------------
IF style% = 1 THEN
  TL$ = " ╔": TR$ = "╗ ": BL$ = " ╚": BR$ = "╝ ": LS$ = "
║": RS$ = "║ " LIN$ = STRING$(WIDE%, "═")
ELSEIF style% = 2 THEN
  TL$ = " ┌": TR$ = "┐ ": BL$ = " └": BR$ = "┘ ": LS$ = "
│": RS$ = "│ " LIN$ = STRING$(WIDE%, " ")
ELSEIF style% = 3 THEN
  TL$ = "  ": TR$ = "  ": BL$ = "  ": BR$ = "  ": LS$ = "
│": RS$ = "│ " LIN$ = STRING$(WIDE%, " ")
END IF
'-------------------------------------------------------------
TOPLINE$ = TL$ + LIN$ + TR$
BOTLINE$ = BL$ + LIN$ + BR$

IF syn$ = "y" OR syn$ = "Y" THEN

COLOR sfc%, sfc%                                  ''''''''''''''
FOR i% = STLR% TO STLR% + TALL% + 1               '
LOCATE i%, STLC%                                  '   shadow
PRINT ; SPC(1);                                   '
NEXT i%                                           ''''''''''''''
PRINT ; SPC(SWIDE% - 1);                             '

END IF

COLOR FGC%, BGC%                                  ''''''''''''''''''
LOCATE TLR%, TLC%                                 '
```

```
PRINT ; TOPLINE$;                       '
FOR i% = TLR% + 1 TO TLR% + TALL%       '
LOCATE i%, TLC%                         '     box
PRINT ; LS$; SPC(WIDE%); RS$;           '
NEXT i%                                 '
LOCATE i%, TLC%                         '
PRINT ; BOTLINE$;                       ''''''''''''''''''
IF bx$ = "y" OR bx$ = "Y" GOTO 9999
j% = TLR% + 1
K% = TLC% + 3
FOR i% = 1 TO LL%
LOCATE j%, K%
PRINT ; item$(i%);
j% = j% + 2
NEXT i%
9999

END SUB

SUB getinput (instr$, maxlen%)
CONST Insert% = 1, overstrike% = 2

'   make required initializations
firstcol% = POS(0)
insertmode% = overstrike%
curpos% = 1

'    display available length with a blue box
PRINT instr$;
COLOR , 1
PRINT SPACE$(maxlen% - LEN(instr$));
LOCATE , firstcol%

'   blink cursor
COLOR 31, 11
IF LEN(instr$) = 0 THEN
   PRINT " "; CHR$(29);
ELSE
   PRINT LEFT$(instr$, 1); CHR$(29);
END IF
COLOR 15, 0

'    get first character
DO
   onechar$ = INKEY$
LOOP WHILE onechar$ = ""

'     process characters one at a time
DO UNTIL onechar$ = CHR$(13)

    '     de-highlight current character
    IF curpos% > maxlen% THEN
```

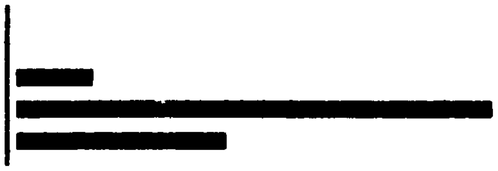
```
            PRINT " "; CHR$(29);
         ELSEIF curpos% > LEN(instr$) THEN
            COLOR 15, 1                                 '?
            PRINT " "; CHR$(29);
            COLOR 15, 0                                   '?
         ELSE
            PRINT MID$(instr$, curpos%, 1); CHR$(29);
         END IF

         '     check for special characters
         IF LEFT$(onechar$, 1) = CHR$(0) THEN

            '     check for <RIGHT>
            IF RIGHT$(onechar$, 1) = CHR$(77) THEN

               '   if before last character, move right one
               IF curpos% <= LEN(instr$) THEN
                  curpos% = curpos% + 1
               END IF

            '     check for <left>
            ELSEIF RIGHT$(onechar$, 1) = CHR$(75) THEN

               '     if past first character, move left one
               IF curpos% > 1 THEN
                  curpos% = curpos% - 1
               END IF

            '    check for <DEL>
            ELSEIF RIGHT$(onechar$, 1) = CHR$(83) THEN

               '   if before last character, remove one
               IF curpos% <= LEN(instr$) THEN
                  instr$ = LEFT$(instr$, curpos% - 1) +
MID$(instr$, curpos% + 1)
                  PRINT MID$(instr$, curpos%); " ";
               END IF
'----------------------------------------
'           check for <INS>
'----------------------------------------
            ELSEIF RIGHT$(onechar$, 1) = CHR$(82) THEN

               '     toggle insert mode
               IF insertmode% = overstrike% THEN
                  insertmode% = Insert%
               ELSE
                  insertmode% = overstrike%
               END IF

            END IF

        ' check for <BACKSPACE>
```

```
   ELSEIF onechar$ = CHR$(8) THEN

      '      if past first character then remove one
      IF curpos% > 1 THEN
         IF curpos% > LEN(instr$) THEN
            PRINT CHR$(29); " ";
         ELSE
            PRINT CHR$(29); MID$(instr$, curpos%); " ";
         END IF
         instr$ = LEFT$(instr$, curpos% - 2) + MID$(instr$,
curpos%)
      curpos% = curpos% - 1
   END IF
'--------------------------------------------------------
'   no special character, so check for overstrike mode
'--------------------------------------------------------
ELSEIF insertmode% = overstrike% THEN

   '   if overstrike, replace that character
   '   or add it to the end
   IF curpos% <= LEN(instr$) THEN
      MID$(instr$, curpos%, 1) = onechar$
      PRINT onechar$;
      curpos% = curpos% + 1
   ELSEIF curpos% <= maxlen% THEN
      instr$ = instr$ + onechar$
      PRINT onechar$;
      curpos% = curpos% + 1
   ELSE
       BEEP
   END IF

'     insertmode must be in insert
ELSEIF curpos% <= maxlen% THEN

   '     insert character, move rest of string over
   instr$ = LEFT$(instr$, curpos% - 1) + onechar$ +
MID$(instr$, curpos%)
   PRINT MID$(instr$, curpos%);
   curpos% = curpos% + 1
ELSE
   BEEP
END IF
'--------------------------------
'   highlight cursor position
'--------------------------------
COLOR 31, 11
LOCATE , firstcol% + curpos% - 1
IF curpos% > LEN(instr$) THEN
   PRINT " "; CHR$(29);
ELSE
   PRINT MID$(instr$, curpos%, 1); CHR$(29);
   END IF
   COLOR 15, 0

   '    get another character
   DO
      onechar$ = INKEY$
   LOOP WHILE onechar$ = ""

LOOP

'   redisplay string without length display
LOCATE , firstcol%
PRINT instr$; SPACE$(maxlen% - LEN(instr$))

END SUB
```

Appendix E: Survey Questions and Descriptive Statistics

1. Being given the opportunity to repeat any exercise as often as necessary enhanced my ability to learn complex mathematical operations.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	0
3.	Undecided	4
4.	Agree	25
5.	S. Agree	11

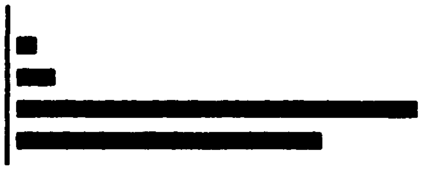


Disagree	- 0%
Undecided	- 10%
Agree	- 90%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.175	5.943E-01	4.000	3.000	5.000

2. The mechanical simplicity of the Matrix Program allowed me to concentrate on LEARNING MATHEMATICAL CONCEPTS rather than on HOW TO USE THE PROGRAM proper.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	1
3.	Undecided	2
4.	Agree	21
5.	S. Agree	16

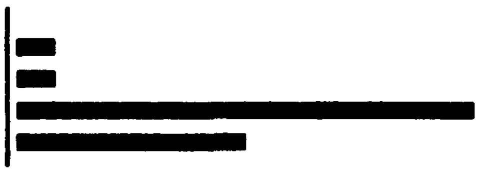


Disagree	- 2.5%
Undecided	- 5.0%
Agree	- 92.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.300	6.869E-01	4.000	2.000	5.000

3. The use of color graphics highlighted critical matrix operations and helped focus my attention on key activities of the process under study.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	2
4.	Agree	24
5.	S. Agree	12



Disagree	-	5%
Undecided	-	5%
Agree	-	90%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.150	7.355E-01	4.000	2.000	5.000

4. Being able to step through an operation at my own pace helped me discover the essential aspects of each matrix operation.

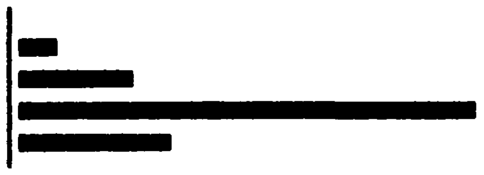
	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	1
3.	Undecided	1
4.	Agree	22
5.	S. Agree	16

Disagree	-	2.5%
Undecided	-	2.5%
Agree	-	95.0%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.325	6.558E-01	4.000	2.000	5.000

5. Mathematical operations SHOULD BE visualized BEFORE any attempt is made to state the process rigorously.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	6
4.	Agree	24
5.	S. Agree	8

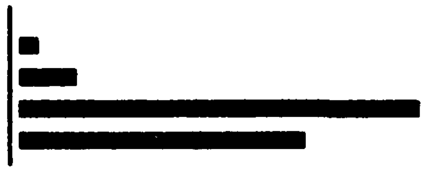


Disagree	-	5%
Undecided	-	15%
Agree	-	80%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.950	7.494E-01	4.000	2.000	5.000

6. Using the MATRIX program was a lot more fun than reading about matrix operations in a text book.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	1
3.	Undecided	3
4.	Agree	21
5.	S. Agree	15

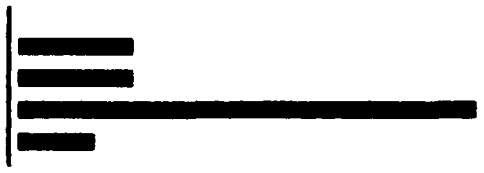


Disagree	-	2.5%
Undecided	-	7.5%
Agree	-	80.0%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.250	7.071E-01	4.000	2.000	5.000

7. The MATRIX program made learning mathematical concepts much easier than being introduced to them via lecture.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	6
3.	Undecided	6
4.	Agree	24
5.	S. Agree	4

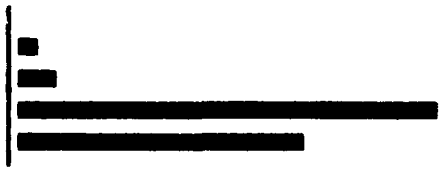


Disagree	-	15%
Undecided	-	15%
Agree	-	70%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.650	8.638E-01	4.000	2.000	5.000

8. Being able to study each matrix operation step by step made it much easier for me to describe the operation verbally.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	1
3.	Undecided	2
4.	Agree	22
5.	S. Agree	15

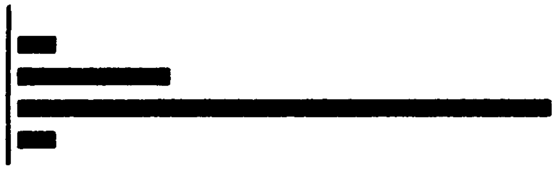


Disagree	-	2.5%
Undecided	-	5.0%
Agree	-	92.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.275	6.789E-01	4.000	2.000	5.000

9. Verbalizing the steps in a matrix operation provided me with an excellent way to display my knowledge about each operation.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	8
4.	Agree	28
5.	S. Agree	2

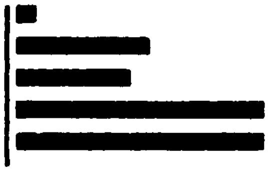


Disagree	- 5%
Undecided	- 2%
Agree	- 75%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.750	6.304E-01	4.000	2.000	5.000

10. Two way communication with an instructor is essential for a student to master the steps involved in performing any matrix operation.

	RESPONSE	FREQUENCY
1.	S. Disagree	1
2.	Disagree	7
3.	Undecided	6
4.	Agree	13
5.	S. Agree	13



Disagree	- 20%
Undecided	- 15%
Agree	- 65%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.750	1.171	4.000	1.000	5.000

11. I prefer to THINK privately about what I know rather than try to objectively display my knowledge by verbalization.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	11
3.	Undecided	7
4.	Agree	15
5.	S. Agree	7

Disagree - 27.5%
Undecided - 17.5%
Agree - 55.0%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.450	1.085	4.000	2.000	5.000

12. It is the teacher's responsibility to ensure I possess an adequate concept base to learn any particular subject.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	16
3.	Undecided	5
4.	Agree	15
5.	S. Agree	4

Disagree - 40.0%
Undecided - 12.5%
Agree - 47.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.175	1.083	3.000	2.000	5.000

13. I don't think verbalization helps me to formulate algorithms at all.

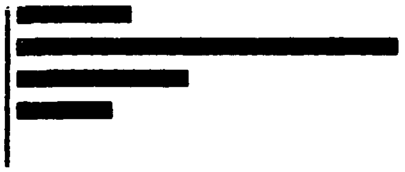
	RESPONSE	FREQUENCY
1.	S. Disagree	3
2.	Disagree	25
3.	Undecided	7
4.	Agree	5
5.	S. Agree	0

Disagree - 70.0%
Undecided - 17.5%
Agree - 12.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
2.350	8.022E-01	2.000	1.000	4.000

14. Anyone can construct an algorithm by observing an operation. There is no need to verbalize what is observed BEFORE constructing such an algorithm.

	RESPONSE	FREQUENCY
1.	S. Disagree	6
2.	Disagree	20
3.	Undecided	9
4.	Agree	5
5.	S. Agree	0

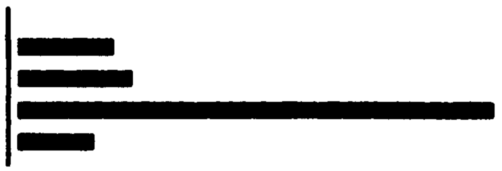


Disagree - 65.0%
Undecided - 22.5%
Agree - 12.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
2.325	8.883E-01	2.000	1.000	4.000

15. Being asked to verbalize the process makes it easier to write an algorithm for the process.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	5
3.	Undecided	6
4.	Agree	25
5.	S. Agree	4

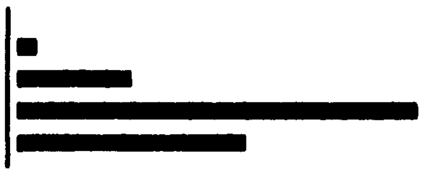


Disagree	-	12.5%
Undecided	-	15.0%
Agree	-	72.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.700	8.228E-01	4.000	2.000	5.000

16. Constructing an algorithm for a matrix operation is an excellent way to rigorously display ones understanding of some process.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	1
3.	Undecided	6
4.	Agree	21
5.	S. Agree	12

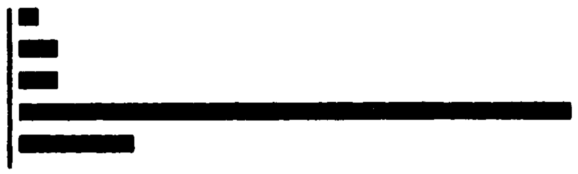


Disagree	-	2.5%
Undecided	-	15.0%
Agree	-	82.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.100	7.442E-01	4.000	2.000	5.000

17. A minimum set of concepts is required to develop an algorithm about any matrix operation.

RESPONSE	FREQUENCY
1. S. Disagree	1
2. Disagree	2
3. Undecided	2
4. Agree	29
5. S. Agree	6

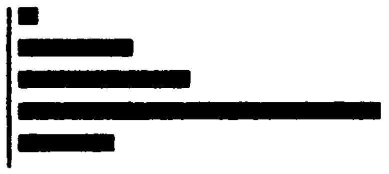


Disagree - 7.5%
Undecided - 5.0%
Agree - 87.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.925	7.970E-01	4.000	1.000	5.000

18. The minimum set of concepts required to develop an algorithm should be provided to the student by the instructor.

RESPONSE	FREQUENCY
1. S. Disagree	1
2. Disagree	6
3. Undecided	9
4. Agree	19
5. S. Agree	5



Disagree - 17.5%
Undecided - 22.5%
Agree - 60.0%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.525	0.987	4.000	1.000	5.000

19. A formal mathematical model of any matrix operation can be constructed without formulating an algorithm for the process.

	RESPONSE	FREQUENCY
1.	S. Disagree	2
2.	Disagree	22
3.	Undecided	6
4.	Agree	10
5.	S. Agree	0

Disagree	- 60%
Undecided	- 15%
Agree	- 25%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
2.600	9.282E-01	2.000	1.000	4.000

20. The mathematical modeling of a matrix operation is facilitated by constructing an algorithm AFTER verbalizing the steps involved in the process under observation.

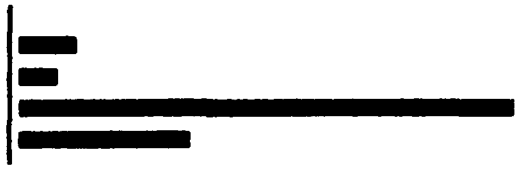
	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	6
4.	Agree	30
5.	S. Agree	2

Disagree	- 5%
Undecided	- 15%
Agree	- 80%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.800	6.076E-01	4.000	2.000	5.000

21. Constructing an algorithm for any matrix operation makes the mathematical modeling of the process much easier.

RESPONSE		FREQUENCY
1.	S. Disagree	0
2.	Disagree	3
3.	Undecided	2
4.	Agree	26
5.	S. Agree	9

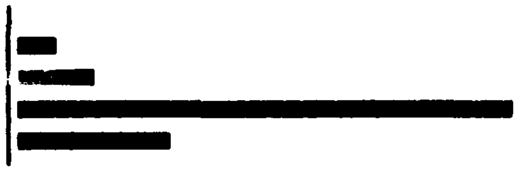


Disagree	- 7.5%
Undecided	- 5.0%
Agree	- 87.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.025	7.675E-01	4.000	2.000	5.000

22. Being asked to construct mathematical models builds a person's confidence in being able to use mathematics in daily work activities.

RESPONSE		FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	4
4.	Agree	26
5.	S. Agree	8

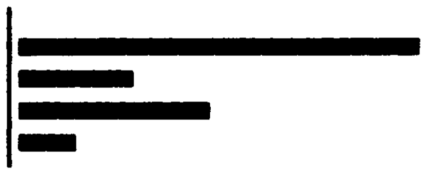


Disagree	- 5%
Undecided	- 10%
Agree	- 85%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.000	7.161E-01	4.000	2.000	5.000

23. The most effective way to communicate a mathematical concept to another person is to present him with the most general form possible -- the mathematical formula.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	21
3.	Undecided	6
4.	Agree	10
5.	S. Agree	3

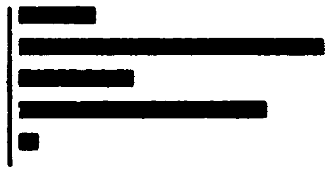


Disagree - 52.5%
Undecided - 15.0%
Agree - 32.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
2.875	1.042	2.000	2.000	5.000

24. Mathematics can be taught without using formal mathematical symbolism.

	RESPONSE	FREQUENCY
1.	S. Disagree	4
2.	Disagree	16
3.	Undecided	6
4.	Agree	13
5.	S. Agree	1

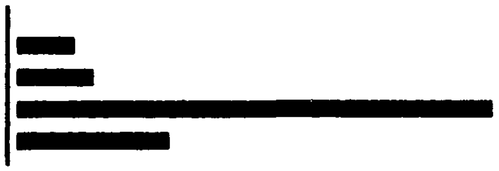


Disagree - 50%
Undecided - 15%
Agree - 35%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
2.775	1.097	2.500	1.000	5.000

25. Before any student is given the mathematical formula for a matrix operation - he/she should have a chance to observe the process and VISUALIZE every step.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	3
3.	Undecided	4
4.	Agree	25
5.	S. Agree	8

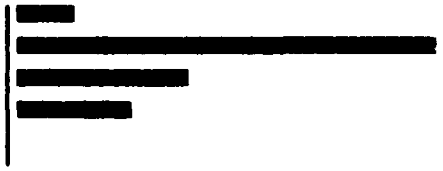


Disagree	-	7.5%
Undecided	-	10.0%
Agree	-	82.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.950	7.828E-01	4.000	2.000	5.000

26. By studying mathematical formulae first, students can grasp what a matrix operation involves without resorting to observing a particular example of the process.

	RESPONSE	FREQUENCY
1.	S. Disagree	3
2.	Disagree	22
3.	Undecided	9
4.	Agree	6
5.	S. Agree	0

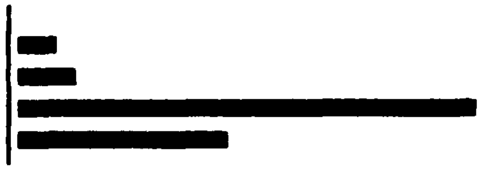


Disagree	-	62.5%
Undecided	-	22.5%
Agree	-	15.0%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
2.450	8.458E-01	2.000	1.000	4.000

27. Using the matrix program as a learning tool was more interesting than listening to an ordinary lecture.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	2
3.	Undecided	3
4.	Agree	24
5.	S. Agree	11

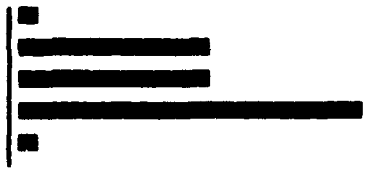


Disagree	- 5.0%
Undecided	- 7.5%
Agree	- 87.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.100	7.442E-01	4.000	2.000	5.000

28. All four steps of the VVAM process should be employed every time a new matrix operation is mastered and mathematically modeled.

	RESPONSE	FREQUENCY
1.	S. Disagree	1
2.	Disagree	10
3.	Undecided	10
4.	Agree	18
5.	S. Agree	1

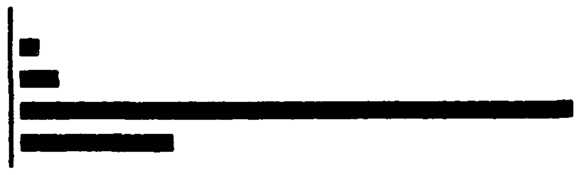


Disagree	- 27.5%
Undecided	- 25.0%
Agree	- 47.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.200	9.392E-01	3.000	1.000	5.000

29. I would use learning tools similar to the MATRIX program if given the opportunity.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	1
3.	Undecided	2
4.	Agree	29
5.	S. Agree	8

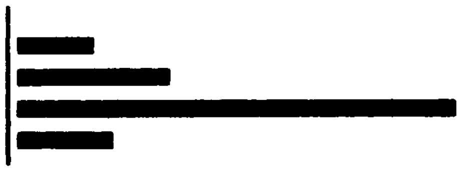


Disagree - 2.5%
Undecided - 5.0%
Agree - 92.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
4.100	5.905E-01	4.000	2.000	5.000

30. A Program like the HANSARD MATRIX SYSTEM should be constructed for all types of mathematical subjects.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	4
3.	Undecided	8
4.	Agree	23
5.	S. Agree	5

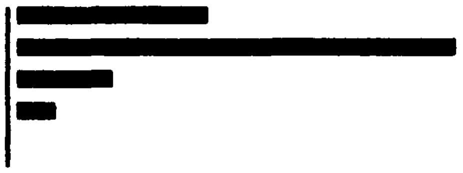


Disagree - 10%
Undecided - 20%
Agree - 70%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.725	8.161E-01	4.000	2.000	5.000

31. I don't think the HANSARD MATRIX SYSTEM helped me learn much about matrix algebra at all.

	RESPONSE	FREQUENCY
1.	S. Disagree	10
2.	Disagree	23
3.	Undecided	5
4.	Agree	2
5.	S. Agree	0



Disagree	- 82.5%
Undecided	- 12.5%
Agree	- 5.0%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
1.975	7.675E-01	2.000	1.000	4.000

32. Using the HANSARD MATRIX SYSTEM motivated me to think deeply about the THEORY of matrix algebra.

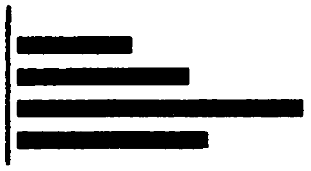
	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	11
3.	Undecided	7
4.	Agree	21
5.	S. Agree	1

Disagree	- 27.5%
Undecided	- 17.5%
Agree	- 55.0%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.300	9.115E-01	4.000	2.000	5.000

33. By using the VVAM heuristic for model building, I became conscious of how mechanical my previous knowledge of mathematics is.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	6
3.	Undecided	9
4.	Agree	15
5.	S. Agree	10

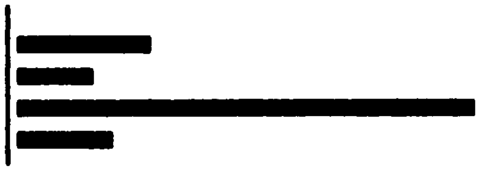


Disagree	- 15.0%
Undecided	- 22.5%
Agree	- 62.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.725	1.012	4.000	2.000	5.000

34. I think the VVAM protocol helped me understand matrix operations better than I could have understood them under typical classroom conditions.

	RESPONSE	FREQUENCY
1.	S. Disagree	0
2.	Disagree	7
3.	Undecided	4
4.	Agree	24
5.	S. Agree	5



Disagree	- 17.5%
Undecided	- 10.0%
Agree	- 72.5%

MEAN	S.D.	MEDIAN	MINIMUM	MAXIMUM
3.675	9.167E-01	4.000	2.000	5.000

Bibliography

1. Novak, Joseph D. and D. Bob Gowin. Learning How to Learn. New York: Cambridge University Press, 1984.

2. Commission on Standards for School Mathematics. *Curriculum and Evaluation Standards for School Mathematics*. Reston VA: National Council of Teachers of Mathematics, 1989.

3. Ausubel, David. Educational Psychology: A Cognitive View. New York: Holt, Rinehart, and Winston, 1968.

4. Banchoff, Thomas. "Student Generated Interactive Software for Calculus of Surfaces in a Workstation Laboratory," UME Trends, 8: 6-7 (August 1989).

5. Novak, Joseph D. A Theory of Education. Ithaca NY: Cornell University Press, 1977.

6. Belcher, Duane M. Giving Psychology Away. San Francisco: Canfield Press, 1973.

7. Emory, William C. Business Research Methods (Third Edition). Homewood IL: Irwin, 1985.

8. Lawrence, Gordon. People Types and Tiger Stripes: A Practical Guide to Learning Styles. Gainesville FL: Center for Applications of Psychological Type, Inc., 1982.

Vita

Captain Stone W. Hansard [redacted]. He graduated from Escambia High School in Pensacola, Florida in 1974, attended Pensacola Junior College on a scholastic scholarship, and continued his education, graduating from the University of West Florida in 1982 with a Bachelor of Science in Systems Science (specialty: Scientific Option). Upon completion of Officer's Training School in 1983 he received his reserve commission and entered the School of Engineering, Air Force Institute of Technology. In 1985 he was awarded a Bachelor of Science in Aeronautical Engineering and began a four year tour at the Air Force Wright Aeronautical Laboratory's Aero Propulsion Lab. In 1986 he received his regular commission. The highlights of his tenure at the Aero Propulsion Laboratory included engineering and development efforts on the Short Range Attack Missile II (SRAM II), the Advanced Medium Range Air-to-Air Missile (AMRAAM), and the National Aero Space Plane (NASP). In May 1989, he entered the School of Systems and Logistics, Air Force Institute of Technology, in the Graduate Systems Management program.

[redacted]

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 1990	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE

AN INTERACTIVE SYSTEM OF COMPUTER GENERATED GRAPHIC DISPLAYS FOR MOTIVATING MEANINGFUL LEARNING OF MATRIX OPERATIONS AND CONCEPTS OF MATRIX ALGEBRA

5. FUNDING NUMBERS

6. AUTHOR(S)

Stone W. Hansard, Capt, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GSM/ENC/90S-12

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This study evaluated students' responses to a teaching method designed to involve students and teachers of mathematics in a meaningful learning experience using a system that allowed the student to participate in discovery learning. The system was comprised of a computer program that motivated visualization of concepts of matrix algebra, and a procedure that encouraged the students to verbalize each concept, create an algorithm for the matrix operation, and discover a mathematical formula that described the process for each operation. The teaching system is based on modern learning theory and reflects several recommendations made by the Commission on Standards for School Mathematics to the National Council of Teachers of Mathematics (NCTM). Of the 40 students who participated in the survey, over 87% felt the matrix program was more interesting than an ordinary lecture, 80% felt the system was more fun than reading a textbook, and 92% indicated they would use a learning tool similar to the matrix computer program if given the opportunity. Over 82% of the students believed the matrix teaching system helped them learn matrix algebra, while 72% felt the teaching system helped them learn matrix algebra better than they would have under typical classroom conditions.

14. SUBJECT TERMS

Learning, Learning Theory, Linear Algebra, Matrix Algebra, Matrix Math

15. NUMBER OF PAGES

185

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL